Robust and Cost-Effective Architecture Design for Smart Grid Communications: A Multi-stage Middleware Deployment Approach

Dong-Hoon Shin, Shibo He, and Junshan Zhang
School of Electrical, Computer and Energy Engineering, Arizona State University
Email: {donghon.shin.2, shibo.he, junshan.zhang}@asu.edu

***Abstract*—Wide-area monitoring, protection and control (WAMPAC) plays a critical role in smart grid, for protection against possible contingencies, by using the Supervisory Control and Data Acquisition (SCADA) system. However, a general consensus is that such a hierarchical system can be highly vulnerable to component (i.e., nodes and links) failures, calling for a *robust* and *cost-effective* communication system for smart grid. To this end, we consider a *middleware* approach to leverage the existing commercial communication infrastructure with abundant connectivity. In this approach, a natural question is how to use the middleware to cohesively "glue" the power grid and the commercial communication infrastructure together, in order to enhance robustness and cost-effectiveness. We tackle this problem while taking into consideration the *multi-stage deployment* of power devices and their redundant connections. We show that this problem can be cast as a *minimum-cost middleware design under incremental deployment*—an "online" problem where the input is provided gradually due to the incremental deployment. We design a randomized "online" algorithm, and show that it achieves the *order-optimal* average competitive ratio. Simulation results demonstrate the performance of our proposed algorithm, compared to the optimal offline solution.**

I. Introduction

Existing power grids are now entering a new era of transformation, driven by today's pressing needs such as cost-effective power generation, higher usage of renewable energy sources and two-way flows of electricity and information. Such a transformation of power grids is expected to put forth *smart grid*, which will enable electricity provisioning in a cleaner, safer, and more reliable and economical way. To accomplish such innovations, the major challenge lies in the reliability and security. This requires smart grid to be equipped with a wide-area monitoring, protection, and control (WAMPAC) system. A WAMPAC system makes use of system-wide information, collected over a wide geographic area, and communications in order to perform fast decision-making and switching actions for the system stability and protection [1]. WAMPAC is playing an increasingly critical role in power systems for their reliable operation and protection from possible contingencies.

Traditionally, the Supervisory Control and Data Acquisition (SCADA) system has been used to monitor and control the power grid. The SCADA system is a star network, where each substation is directly connected to a control center, and operates in a centralized fashion, i.e., the control center gathers data from sensors and sends commands to control devices. Due to the centralized structure, however, the SCADA system is not scalable, and is thus used for only local monitoring and control. For WAMPAC, the SCADA system has been extended to a hierarchical system with a few levels of local centers. For instance, the Bonneville Power Administration, based in the Pacific Northwest, has built a phasor network with a hierarchy of three levels [2]: at the first level, Phasor Measurement Units (PMUs) are dispersed throughout the power grid; at the second level, Phasor Data Concentrators (PDCs) are deployed to collect data at a local level; at the third level, the SCADA system is employed at the control center.

Such a hierarchical WAMPAC system is efficient for data aggregation and command delivery. On the flip side of the coin, it can be highly vulnerable to component (i.e., node and link) failures, since it relies on a small number of critical nodes (i.e., PDCs or Super-PDCs) and their associated links for the entire system to function. To enhance the robustness, one might consider increasing the redundancy of high-level nodes and their links, thereby improving the connectivity between the remote devices (i.e., sensors and actuators) and the control center. However, it would be too costly to build such a dedicated communication network providing multiple redundant connections to a large number of remote devices. These vulnerabilities and limitations of the existing WAMPAC systems motivate a need for a robust and cost-effective communication system for smart grid.

To this end, we consider an alternative approach by utilizing the existing commercial communication infrastructure with abundant connectivity, rather than building a dedicated communication infrastructure for smart grid. Nevertheless, a challenge in this approach is that the existing commercial communication infrastructures are not designed to meet the data delivery requirements for smart grid, such as heterogeneous delivery rates and latencies, reliability, and security [3], [4]. Also, the communication system for smart grid should be flexible, so that it can deal with heterogeneous communication technologies[1] of the sub-systems of smart grid. To address

[1]Smart grids will rely on various existing and future communication networking technologies, such as power line communications (PLC), fiber-optic communications, short-range wireless communication technologies (e.g., Wi-Fi and ZigBee), cellular and broadband wireless technologies (e.g., 3G, WiMax and LTE), and IP-based networks [5].

This research was supported in part by the U.S. DTRA grant HDTRA1-13-1-0029 and National Science Foundation under grants CPS-1035906.

978-1-4799-3360-0/14/$31.00 ©2014 IEEE

these issues, we advocate to deploy *middleware*[2] on top of the commercial communication infrastructures to be utilized for the smart grid communications. The middleware will function as an interface between the power applications and the commercial communication infrastructures, thus offering a great flexility to the smart grid communications system. Further, the middleware will allow the system administrator to efficiently manage the underlying network resources, so that various data delivery requirements for power applications can be satisfied [6]–[8].

With this approach, a natural question to ask is how to "glue" the power grid and the commercial communication infrastructure together, by using the middleware (i.e., middleware gateways and communication links connecting the two systems), so as to build a *robust* and *cost-effective* smart grid communications system. This problem boils down to the following: Where to strategically deploy middleware in the commercial communication infrastructures, such that a desired number of redundant connections between the remote devices and the control center is provided while minimizing the middleware deployment cost. In reality, there are often cases in which remote devices and redundant connections are deployed *incrementally*, such as PMUs being gradually installed at substations. This happens for a variety of reasons, including sensors/actuators added on demand, gradually increased redundant connections, and limited budget on purchasing devices and establishing connections. Thus, in practice, while the system administrator can know the *potential* locations at which remote devices are deployed, he or she may not know in advance whether they will be indeed deployed at the locations and also how many redundant connections will be requested. Therefore, the system administrator needs to make judicious decisions on the middleware deployment, so that the connection requests can be satisfied not only for immediate ones but also for ones that may arrive in future.

In this paper, we tackle this challenging problem. Our main contributions are summarized as follows:

- We propose a *middleware based robust and cost-effective architecture* for smart grid communications. To this end, we formulate the *minimum-cost middleware design under incremental deployment* (MCMD-ID) problem. MCMD-ID is an "online" problem, where the input is provided gradually. We prove that, for MCMD-ID, no "online" algorithm[3] can achieve a *competitive ratio* better (i.e., smaller) than $O(\log N \log M)$, where N is the minimum number of connections required to meet all redundancy requirements and M is the number of commercial network nodes over which middleware is (potentially) deployed.
- We design an *order-optimal* competitive randomized algorithm for MCMD-ID, which achieves an expected competitive ratio of $O(\log N \log M)$[4]. To this end, we first transform MCMD-ID into an equivalent problem, called the *online minimum-cost network design* (OMCND) problem. We then design an $O(\log N \log M)$-competitive randomized online algorithm for OMCND. For this, we first develop an $O(\log M)$-competitive deterministic algorithm to compute a near-optimal fractional solution for OMCND. We next devise a randomized online rounding procedure to convert the fractional solution to an integer solution feasible for OMCND, which employs a randomized rounding followed by a remedy procedure to meet the demands unsatisfied after the randomized rounding. Finally, we develop a greedy post-processing to further optimize the solution.

The design of our online algorithm for OMCND is inspired by the online LP rounding based algorithm in [9], which solves an online problem called *multicast problem*. OMCND is a variant of the multicast problem with two generalizations—allowing each demand (i.e., each client in the multicast problem) to have multiple connections to the root of the tree, and to be associated with multiple leaf nodes in each subtree. Due to these generalizations, the online algorithm for the multicast problem in [9] cannot be applied to our problem, OMCND. We design our online algorithm for OMCND to integrate the two generalizations made in OMCND. Also, our competitive-ratio analysis of the proposed algorithm is very different in details from the one in [9] due to the aforementioned reasons.

II. Problem Formulation and Hardness

A. Minimum-Cost Middleware Design under Incremental Deployment

Consider a graph $G = (P \cup Q, E_Q \cup E_{P\text{-}Q})$: P denotes a set of nodes (potentially or already deployed) in the power system; Q denotes a set of nodes in the commercial network over which a set of middleware gateways will be deployed; E_Q and $E_{P\text{-}Q}$ represent a set of (intra-)edges connecting two network nodes in Q and a set of (inter-)edges that can potentially connect a power node in P and a middleware gateway to be deployed on a network node in Q, respectively. The set of power nodes, P, consists of a control center, p_c, and a set of remote devices (e.g., sensors and actuators), P_r. We are given cost functions $c_v : Q \to \mathbb{R}_+$ and $c_l : E_{P\text{-}Q} \to \mathbb{R}_+$, where $\mathbb{R}_+$ is the set of non-negative real numbers; $c_v(q)$ and $c_l(p,q)$ represent the cost of deploying a middleware gateway on the network node q and the cost of establishing a communication link between the power node p and the middleware gateway deployed on the network node q, respectively. These costs will vary depending on the target applications, types of communications and networking technologies, the distances between power nodes and commercial network nodes, etc.

[2]Middleware is a layer of software above the operating system that provides higher-level building blocks for programmers to use [6].

[3]An online algorithm is one that can process its input piece-by-piece in a sequential manner, i.e., in the order that the input is given to the algorithm, without having the entire input available from the beginning.

[4]This result is obtained under a reasonable assumption that the maximum inter-edge degree of network nodes is bounded by a constant. However, even without this assumption, our proposed algorithm still achieves the same order of competitive ratio, i.e., $O(\log^2(\cdot))$. See Corollary 1 in Section V-B for the details, and Definition 1 in Section II-B for the definition of competitive ratio.

The remote devices in P_r are partitioned into *groups* such that a different redundancy of connectivity should be met for each group of remote devices. That is, each group H will be given a redundancy of connectivity, say r_H, so that even if r_H components of nodes and links fail in the network, there would still exist at least one path of functioning components between a remote device in H and the control center. Typically, r_H would take a small number, say at most three[5]. Note that the notion of "groups" of remote devices allows to capture a variety of realistic application scenarios. For example, in cases where each actuator is critical to the operation of the power system, the system administrator can form groups such that each group contains a single actuator and provide a desired number of additional connections to each actuator. Or, there might be cases where sensors are divided into clusters such that the measurements of the sensors in each cluster provide redundant information and the system administrator has a limited budget on the expenses of the middleware deployment. In such cases, the system administrator can form groups such that each group contains the sensors in a cluster and provide only one redundant connection to each group of sensors. We denote by $G[Q]$ the subgraph of G induced by Q, which represents the commercial network. We assume that $G[Q]$ has abundant connectivity so that $G[Q]$ can provide at least $r_{\max} + 1$ disjoint paths to any pair of nodes in it, where $r_{\max} \triangleq \max_{\forall H} r_H$.

We are particularly interested in *incremental deployment*, where remote devices are deployed incrementally in the power system (while the control center is already placed before remote devices are deployed) and also redundant connections for the groups of remote devices may be added gradually. Specifically, a subset (or the entire set) of remote devices, unknown in advance, are deployed step by step, and which remote devices are deployed in each step is known only at the step. Also, the redundancy of connectivity for each group, i.e., r_H, may increase step by step. Given a (set of) new remote device(s) or an increased redundancy of connectivity for a group, we must *immediately* provide the required number of connections to the (set or group of) remote device(s) via the commercial network by deploying middleware. That is, we are not allowed to wait until other new remote devices or redundancy requirements arrive. This requires middleware to be deployed in *multiple stages*. Our objective is to meet each of the connection requests upon arrival, by judiciously choosing a subset of network nodes in Q (to deploy middleware gateways over) and also a subset of inter-edges in $E_{P\text{-}Q}$ (to connect the middleware gateways and the power nodes), while minimizing the cost of deploying the middleware gateways and establishing the inter-edges.

We recast the objective by taking into account the abundant connectivity that the commercial network has (i.e., any pair of nodes in $G[Q]$ has at least $r_{\max} + 1$ disjoint paths). For notational convenience, we abuse the term "group" to also indicate a subset of remote devices newly deployed, and define H_c as a group associated with the control center p_c and $r_{H_c} \triangleq r_{\max}$. Let $\mathcal{H}$ be the set of all groups. Then, the objective can be recast as the following: upon arrival of each new connection request for a group of remote devices, $H \in \mathcal{H}$, add nodes in Q to Q' and inter-edges in $E_{P\text{-}Q}$ to E' such that two groups H and H_c are connected to at least r_H+1 nodes and at least $r_{H_c}+1$ nodes in Q', respectively, by the inter-edges in E', while minimizing the cost of Q' and E'. We refer to this problem as the ***minimum-cost middleware design under incremental deployment*** (MCMD-ID) problem. (Refer to Fig. 1(a) for an instance of MCMD-ID.)

B. Hardness of MCMD-ID

MCMD-ID is an *online problem* where the input is revealed gradually. Therefore, we need an *online algorithm*, which makes a decision upon arrival of each new input and never revokes the previously-made decisions. A standard measure to evaluate the performance of online algorithms is the "competitive ratio", which compares the performance of an online algorithm to that of the offline algorithm that is given the whole input sequence beforehand [11]. We formally define the competitive ratio as follows.

***Definition** 1 ([11]):* For any instance $\mathcal{I}$ of a minimization problem, we say that an online algorithm $\mathcal{A}$ is γ*-competitive*, or achieves a *competitive ratio* of γ, if $\mathcal{A}$ yields a solution of cost at most $\gamma \cdot \mathrm{OPT}(I) + \delta$, where $\mathrm{OPT}(I)$ is the optimal cost of an offline algorithm for the instance I and the additive term δ is a constant independent of the sequence of the input.

We have a lower bound on the competitive ratio of any online algorithm for MCMD-ID.

***Theorem** 1:* Any deterministic (or randomized) algorithm for MCMD-ID has a competitive ratio of at least $\Omega(\log N \log M)$ unless P = NP (or unless NP $\subseteq$ BPP[6]), where N is the minimum number of connections required to meet all demands and M is the number of commercial network nodes.

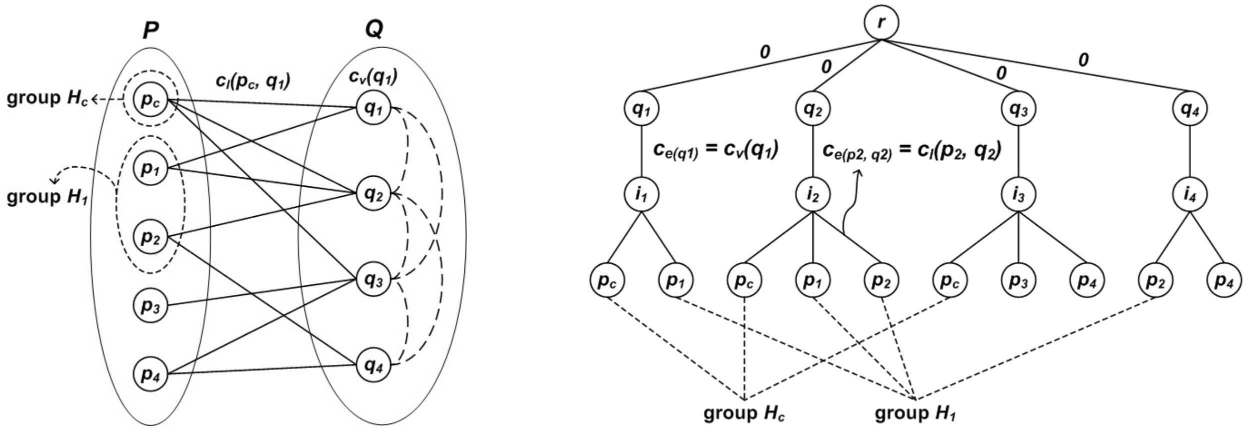
To prove the theorem, one can show a reduction from the online set cover problem [12] to MCMD-ID, and then use a lower bound on the competitive ratio for the online set cover problem, known in [13]. We omit the proof due to space limitation, and make it available in [14].

III. From MCMD-ID to OMCND and Online Algorithm

In this section, for ease of exposition, we transform MCMD-ID into an equivalent problem, called the *online minimum-cost network design* (OMCND) problem. We then give a summary of the proposed online algorithm for OMCND.

[5]Conventional U.S. power system planning is typically performed with the consideration of possible contingencies of intrinsic equipment failure, where one component (or one set of closely related components) fails. Occasionally, some regions of the U.S. power system are designed to handle a few contingencies, i.e., simultaneous failures of 2 or 3 components. [10]

[6]BPP (bounded-error probabilistic polynomial time) is the class of decision problems solvable by a probabilistic Turing machine in polynomial time, with an error probability of at most $\frac{1}{3}$ for all instances. The relationship between NP and BPP is unknown. However, it is conjectured in the literature that P = BPP $\subseteq$ NP, since P = BPP under some reasonable assumptions.



(a) An instance of MCMD-ID

(b) The instance of OCMND equivalent to that of MCMD-ID in Fig. 1(a).

Fig. 1. An illustrative example of MCMD-ID and its equivalent transformation into OCMND.

A. Transformation of MCMD-ID into OMCND

Let T be a rooted tree with the root r (see Fig. 1(b)). It consists of a set of subtrees of depth two, $\mathcal{T} = \{T_q : q \in Q\}$, each having the root connected to r. Each subtree $T_q \in \mathcal{T}$, associated with a network node $q \in Q$, consists of the root, one internal node and a set of leaf nodes, each corresponding to a power node in P (i.e., the control center or a remote device) connected to the network node q by an inter-edge. Let E_{T_q} be the set of all edges in T_q and E be the set of all edges in T. Each edge $e \in E$ is assigned a cost c_e defined as follows. For an edge that connects r and the root of a subtree, i.e., $e \in E - \cup_{q \in Q} E_{T_q}$, let $c_e = 0$. For an edge $e \in E_{T_q}$, c_e is defined as the following: for the edge incident to the root of T_q, denoted by $e(q)$, $c_{e(q)} = c_v(q)$; for an edge incident to a leaf node of T_q corresponding to a power node p, denoted by $e(p,q)$, then $c_{e(p,q)} = c_l(p,q)$.

Let $D = (d_1, d_2, \dots)$ be a sequence of demands, each having the form of $d_i = (L_i, k_i)$ where L_i is a set of leaf nodes in T and k_i is a positive integer. Each demand is created upon arrival of a connection request as follows. Given a connection request for a group $H \in \mathcal{H}$ of remote devices with r_H, we first create a demand $d_i = (L_i, k_i)$ such that L_i is the set of the leaf nodes, each corresponding to a remote device in H, and $k_i = r_H + 1$. If $k_i > \max_{j \in \{0,1,\dots,i-1\}} k_j$ where $k_0 \triangleq 0$, right after d_i, we create another demand, $d_{i+1} = (L_{i+1}, k_{i+1})$, which is associated with the control center p_c, such that L_{i+1} is the set of the leaf nodes corresponding to p_c and $k_{i+1} = k_i$. Define a *binary-valued* weight function $w_b : E \to \{0,1\}$. We say that a demand d_i is *satisfied* by w_b if the edges chosen by w_b provide at least k_i *disjoint* paths from r to the leaf nodes in L_i, each path connecting a leaf node in a different subtree to the root r. Let $c(w_b) \triangleq \sum_{e \in E} c_e w_b(e)$ denote the cost of w_b. Our objective is to not only satisfy the new demand(s) upon arrival but also minimize the cost $c(w_b)$ by judiciously updating w_b in a *non-decreasing* fashion, i.e., whose weights cannot be decreased to meet the new demand(s) d_i (and d_{i+1}). We refer to this problem as the ***online minimum-cost network design*** (OMCND) problem.

To show the equivalence between MCMD-ID and OMCND, we consider a feasible solution w'_b to OMCND. We construct (Q', E') from w'_b as the following: if $w'_b(e(q)) = 1$, then $q \in Q'$; if $w'_b(e(p,q)) = 1$, then $(p,q) \in E'$. Clearly, all demands up to d_i (and d_{i+1}) are satisfied by w'_b if and only if two groups H and H_c are connected to at least $r_H + 1$ nodes and at least $r_{H_c} + 1$ nodes in Q', respectively, by the edges in E'. Also, the cost $c(w'_b)$ is equal to the cost of (Q', E'). Hence, we obtain the following theorem.

***Theorem** 2:* MCMD-ID is equivalent to OMCND.

B. Summary of Proposed Online Algorithm for OMCND

Our algorithm for OMCND is an online LP rounding algorithm, based on the one in [9]. It consists of three components (see Fig. 2): 1) online algorithm to compute a fractional solution to OMCND (OAFS-OMCND); 2) randomized online rounding algorithm to obtain an integer solution feasible to OMCND (RORA-OMCND); 3) greedy post-processing (GPP) to further optimize the integer solution.

In the first part, we compute a near-optimal *fractional* solution to OMCND. By the fractional solution, we mean a fractional weight function $w_f : E \to [0,1]$ such that, for each demand $d_i = (L_i, k_i)$, a flow of at least k_i can be sent from the root r to L_i without exceeding the weight (i.e., capacity) of each edge assigned by w_f. We compute w_f in an *online* manner, i.e., such that the fractional weights in w_f are not decreased while meeting each new demand. This online-manner update of w_f is needed to ensure the randomized rounding in the second part not to remove the chosen edges. The fractional solution w_f is guaranteed to have a cost at most $O(\log |E|)$ times the cost of the optimal solution.

In the second part, we convert the fractional solution w_f to an integer solution feasible for OMCND. To this end, we first round the fractional solution in a probabilistic manner, and then make an economical remedy to the (fractional) solution in case when the demand is not satisfied after the probabilistic rounding. The probabilistic rounding can meet the demands with a high probability by paying only a small expected cost, i.e., at most $O(\alpha_f \log |E| \log N)$, where α_f is the cost of the

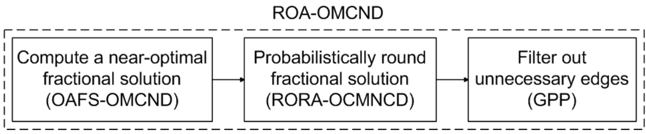


Fig. 2. Three components of ROA-OMCND.

optimal (offline) solution and N is the minimum number of connections required to meet all demands. Also, the remedy procedure incurs only a negligible expected cost, i.e., at most $O(\alpha_f \log N)$. Thus, the overall expected cost of the solution is $O(\alpha_f \log |E| \log N)$, which is the best that one can hope for OMCND (see Theorem 1).

In the third part, we filter out unnecessary edges chosen by the probabilistic rounding. For this, we perform a greedy post-processing, which picks only the disjoint paths of the edges chosen by the integer solution that require the minimum additional cost to meet each new demand.

IV. Randomized Online Algorithm for OMCND

In this section, we design a randomized online algorithm for OMCND that achieves the *order-optimal* expected competitive ratio, i.e., $O(\log N \log M)$. We first describe how to obtain a near-optimal fractional solution to OMCND (in Subsection IV-A). We then explain how to convert the fractional solution to an $O(\log N \log M)$-competitive integer solution to OMCND (in Subsection IV-B). Finally, we present the greedy post processing which further optimizes the solution (in Subsection IV-C).

For ease of exposition, we assume that OMCND has a feasible solution. We can easily verify this by choosing all edges in E and then checking if the demands can be satisfied.

A. Online Algorithm for Computing a Near-Optimal Fractional Solution to OMCND

We denote an optimal (offline) fractional solution by w_f^* and let $\alpha_f = \sum_{e\in E} c_e w_f^*(e)$ be the cost of w_f^*. First, we assume that the value of α_f is known up to a factor of 2 by employing the doubling technique [9]. Initially, we guess $\alpha_f = \min_{e\in E} c_e$, and then run the algorithm using this value as a lower bound on α_f. During the execution of the algorithm, it may turn out that we cannot meet some demand, or α_f is larger than our current guess for α_f. The latter case can be verified by checking if the cost of the fractional solution exceeds an upper bound on the cost of the algorithm, which is $4\alpha_f \log_2 |E| + 2\alpha_f + 2$, known through the competitive-ratio analysis of the algorithm (Theorem 3). If either of the two cases happens, we can "forget" about all weights assigned so far to the edges, double the current guess for α_f, and reiterate the algorithm (see Alg. 2). By forgetting about the weights, we mean that in any iteration, the algorithm only uses the current set of weights in order to meet the demands. However, at any point of time, the actual weight of each edge is the maximal weight assigned to it in any iteration up to the current one. This ensures that the edges previously chosen will not be unselected over the iterations. Note that the cost of the fractional weights that we have forgotten about can increase the cost of our solution by at most a factor of 2, since the value of α_f is doubled at each step. Also, note that the guessed value for α_f grows exponentially fast. Thus, the number of iterations due to the doubling is at most $\log_2(\frac{\max_e c_e}{\min_e c_e}|E|)$.

We can thus assume that α_f is known. We choose all edges whose cost is less than $\frac{\alpha_f}{|E|}$, i.e., $w_f(e) = 1$ if $c_e < \frac{\alpha_f}{|E|}$, by paying a cost of at most α_f. Also, we choose none of the edges whose cost exceeds α_f, i.e., $w_f(e) = 0$ if $c_e > \frac{\alpha_f}{|E|}$, since the optimal fractional solution, w_f^*, assigns zero weight to those edges (refer to [14] for the proof of this claim). Thus, we can assume that all edges have a cost between $\frac{\alpha_f}{|E|}$ and α_f. Further, we normalize the costs by $\frac{\alpha_f}{|E|}$. This makes all costs between 1 and $|E|$. Thus, we only need to update the weights of the other edges, i.e., edges e with $1 \le c_e \le \alpha_f$. Initially, we assign them the same fractional weight of $\frac{1}{|E|^2}$, which makes the total initial cost less than 1.

For the purpose of the analysis, we define a set of dummy variables, w'_e for each edge $e \in E$, and initialize them by letting $w'_e = w_f(e)$ for all $e \in E$. Let $x(L_i; w_f)$ be the value of the maximum flow from the root r to L_i that can be sent without exceeding the weight of each edge given by w_f. We define a "r-L_i cut" as a set of edges that can disconnect r from the leaf nodes in L_i when they are removed. We can easily find $x(L_i; w_f)$ (or a r-L_i cut of the minimum weight) by iteratively searching the maximum flow (or the minimum-weight cut) between r and the leaf nodes of L_i in each subtree.

Upon arrival of each new demand d_i, we first check if $x(L_i; w_f) \ge k_i$. If this is true, then we do nothing since the current fractional weights of w_f already satisfy the demand d_i. Otherwise, i.e.,. if $x(L_i; w_f) < k_i$, we increment the fractional weights of w_f as follows:

Step 1. Find a r-L_i cut of the minimum weight, C_i.

Step 2. For each $e \in C_i$ with $w_f(e) < 1$, let $w'_e \leftarrow w'_e(1+\frac{1}{c_e})$ and $w_f(e) \leftarrow \min\{1, w'_e\}$.

We refer to this algorithm as the *online algorithm for a fractional solution to OMCND* (OAFS-OMCND). As shall be shown later (Theorem 3), for any sequence of demands, the cost of OAFS-OMCND never exceeds $4\alpha_f \log_2 |E| + 2\alpha_f + 2$.

B. Order-Optimal Competitive Randomized Online Rounding Algorithm for OMCND

We round the fractional solution obtained by OAFS-OMCND in two steps. In the first step, we round the fractional solution in a probabilistic manner that satisfies each new demand with a high probability, while paying a small expected cost. In the second step, we meet the demands unsatisfied at the first step by providing the cheapest disjoint paths, which incur an insignificant average cost. To accomplish this, we employ the randomized rounding (RR) method in [9] for the first step, and develop an algorithm called *minimum-cost disjoint path searching* (MCDPS) algorithm for the second step. We describe the procedures of RR and MCDPS below, and present in Alg. 1 how to integrate them with OAFS-OMCND in order to yield an $O(\log N \log M)$-competitive

Algorithm 1 Randomized Online Rounding Algorithm for OMCND (RORA-OMCND)

1: // α_f is given by Alg. 2
2: **if** $i = 1$ **then**
3: $\quad$ $(\{c_e\}, w_f) \leftarrow \texttt{Init}(\{c_e\}, \alpha_f)$
4: $\quad$ $w_b(e) \leftarrow 0$ for all $e \in E$
5: **end if**
6: $M \leftarrow$ 'Success'
7: **while** $n(L_i; w_b) < k_i$ **do**
8: $\quad$ **while** $x(L_i; w_f) < n(L_i; w_b) + 1$ **do**
9: $\quad\quad$ Find a r-L_i cut of the minimum weight, C_i
10: $\quad\quad$ **for** each $e \in C_i$ with $w_f(e) < 1$ **do**
11: $\quad\quad\quad$ $w_f(e) \leftarrow \min\left\{1, w_f(e)\left(1 + \frac{1}{c_e}\right)\right\}$
12: $\quad\quad$ **end for**
13: $\quad\quad$ **if** $\sum_{e \in E} c_e w_f(e) > 2|E|\left(2\log_2|E| + 1 + \frac{1}{\alpha_f}\right)$ or no change occurred to the weights of w_f during the weight increment **then**
14: $\quad\quad\quad$ // i.e., if the current guess for α_f is wrong
15: $\quad\quad\quad$ $M \leftarrow$ 'Error'; then terminate
16: $\quad\quad$ **end if**
17: $\quad$ **end while**
18: $\quad$ $w_b \leftarrow \texttt{RR}(w_f, w_b)$
19: $\quad$ **if** no change occurred to the value of $n(L_i; w_b)$ before and after the execution of RR **then**
20: $\quad\quad$ $E_a \leftarrow \texttt{MCDPS}(w_b)$
21: $\quad\quad$ $w_b(e) \leftarrow 1$ for all $e \in E_a$
22: $\quad$ **end if**
23: **end while**
24: **return** w_f, w_b, M

solution to OMCND. We refer to Alg. 1 as the *randomized online rounding algorithm for OMCND* (RORA-OMNCD).

Randomized Rounding (RR). Given a fractional weight function w_f and a binary weight function w_b (maintained in Alg. 1), the algorithm updates the weights of w_b as follows. Let n be the minimum number of leaf nodes required to be connected to the root r to meet all demands that have arrived so far. For each subtree T_q, the algorithm keeps $2\lceil\log_2(n+1)\rceil$ independent random variables, $X(T_q, j)$, $j = 1, \ldots, 2\lceil\log_2(n+1)\rceil$, uniformly distributed in the interval $[0, 1]$. Each subtree T_q is associated with a threshold $\theta(T_q) \triangleq \min\{X(T_q, j) : j = 1, \ldots, 2\lceil\log_2(n+1)\rceil\}$. For each $e \in E$, let $w_b(e) = 1$ if $w_f(e) > \theta(T_e)$, where T_e denotes the subtree containing edge e.

Minimum-Cost Disjoint Path Searching (MCDPS). Given a binary weight function w_b (maintained in Alg. 1) and letting E_{w_b} be the set of edges chosen by w_b, the algorithm finds a set of edges, $E_a \subseteq E - E_{w_b}$, such that: adding the edges in E_a to E_{w_b} provide at least one additional disjoint path from the root r to L_i while minimizing the cost of the additional edges, i.e., the cost of the edges in E_a. To obtain such an E_a, at each iteration, the algorithm focuses on one of the paths from a leaf node in L_i to the root r, and computes the

Algorithm 2 Randomized Online Algorithm for OMCND (ROA-OMCND)

1: // Given $d_i = (L_i, k_i)$, performs the following procedure
2: **if** $i = 1$ **then**
3: $\quad$ $\alpha_f \leftarrow \min_{e \in E} c_e$
4: $\quad$ $w_b^{\max}(e) \leftarrow 0$ for all $e \in E$
5: $\quad$ $E^{\text{SOL}} \leftarrow \emptyset$
6: **end if**
7: $(w_f, w_b, M) \leftarrow \texttt{RORA-OMCND}(\{c_e\}, w_f, w_b, d_i, \alpha_f)$
8: **while** $M =$ 'Error' **do**
9: $\quad$ // i.e., if our current guess for α is wrong
10: $\quad$ $\alpha_f \leftarrow 2 \cdot \alpha_f$
11: $\quad$ **for** $j \leftarrow 1$ to i **do**
12: $\quad\quad$ $(w_f, w_b, M) \leftarrow \texttt{RORA-OMCND}(\{c_e\}, w_f, w_b, d_j, \alpha_f)$
13: $\quad\quad$ **if** $M =$ 'Error' **then**
14: $\quad\quad\quad$ Break $\quad$ // i.e., go to Line 10
15: $\quad\quad$ **end if**
16: $\quad$ **end for**
17: **end while**
18: $w_b^{\max}(e) \leftarrow \max\{w_b^{\max}(e), w_b(e)\}$ for all $e \in E$
19: $E^{\text{SOL}} \leftarrow \texttt{GPP}(w_b^{\max}, E^{\text{SOL}})$
20: **return** E^{SOL}

additional cost to provide the path. Among them, the algorithm finds a path that requires the minimum additional cost and are disjoint from the paths of the edges in E_{w_b}. Finally, it includes in E_a only the edges in the chosen path that are not in E_{w_b}.

In Alg 1, the notation $n(L_i; w_b)$ (in Lines 7 and 8) represents the number of leaf nodes in L_i connected to the root r. The algorithm verifies that the guessed value for α_f is valid by checking the condition in Line 13 (note that the cost is denormalized). It outputs a message $M =$ 'Error' if the current guess for α_f is not valid. Note that none of the weights of w_b decrease over iterations. This is because as n increases, in RR, each subtree maintains a larger number of random variables, and the thresholds, $\theta(T_q)$, would not decrease due to the increase of the random variables. Also, MCDPS only adds edges to meet the demands. Clearly, the algorithm will terminate, and yield a feasible solution w_b for OMCND if $M =$ 'Success'.

C. Greedy Post-Processing

Note that the solution yielded by RORA-OCMND may include unnecessary edges due to the randomized rounding. Hence, we need to filter out those unnecessary edges in order to reduce the cost of the solution. To this end, we develop a greedy post-processing (GPP) described below. We also present the overall algorithm in Alg. 2, which we refer to as the *randomized online algorithm for OMCND* (ROA-OMCND).

Greedy Post-Processing (GPP). Given a binary weight function w_b (maintained as $w_b^{\max}$ in Alg. 2) and a solution E^{SOL} satisfying all previous demands, the algorithm adds a set of edges, E_a, to E^{SOL} such that $E^{\text{SOL}} \cup E_a$ satisfies the demand while minimizing the cost of the edges in E_a. The algorithm finds such an E_a by the following procedure. Let n_i

be the minimum number of leaf nodes in L_i required to meet the new demand d_i. At each iteration, the algorithm focuses on a path from a leaf node in L_i to the root r, and computes the additional cost required to provide the path. Among those paths, it finds the n_i paths that require the minimum additional cost and are disjoint from the paths of edges in E^{SOL} that connect a leaf node in L_i to the root r. Finally, it includes in E_a only the edges in the n_i chosen paths that are not in E_{w_b}.

V. Competitive-Ratio Analysis

In this section, we analyze the competitive ratio of ROA-OMCND. In Subsection V-A, we show an upper bound on the cost of OAFS-OMCND. In Subsection V-B, we analyze the cost of ROA-OMCND using the obtained upper bound on the cost of OAFS-OMCND.

A. Analysis of the Expected Cost of OAFS-OMCND

In this subsection, we prove the following theorem.

***Theorem** 3:* The cost of the fractional solution yielded by OAFS-OMCND is at most $4\alpha_f \log_2 |E| + 2\alpha_f + 2$.

To prove the theorem, we first assume that the value of α_f is fixed, and compute an upper bound on the cost of OAFS-OMCND. We then multiply the upper bound by 2, since the doubling technique allows us to know α_f up to only a factor of 2. To begin with, we define the following potential function:

$$\Phi \triangleq \sum_{e\in E} c_e w_f^*(e) \log_2 w'_e,$$

where w_f^* denotes an optimal fractional solution to OCMND. Consider an iteration of the algorithm when demand $d_i = (L_i, k_i)$ is given. Let $C'_i \subseteq C_i$ be a subset of edges in the cut C_i whose weight is less than 1, and let $l = |C_i| - |C'_i|$. Denote by $\Delta\Phi$ and $\Delta c(w_f)$ the increments of Φ and $c(w_f)$, respectively, in a single iteration of the weight increment (i.e., Steps 1 and 2 of OAFS-OMCND). We have a lower bound on $\Delta\Phi$ as shown in the following lemma.

***Lemma** 1:* $\Delta\Phi \geq k_i - l$.

Proof: Recall that C_i is a r-L_i cut of the *minimum weight*. This implies that C_i can contain at most one edge of weight 1 from each subtree. Since C_i contains l edges with $w'_e = 1$, C'_i must contain its edges from $|Q| - l$ subtrees. Since w_f^* has to meet demand d_i, w_f^* must allow a flow of at least k_i from the root r to L_i. This, along with the max-flow min-cut theorem, implies that $\sum_{e\in C'_i} w_f^*(e) \geq k_i - l$. Also, it is true that $(1 + 1/c_e)^{c_e} \geq 2$ for $c_e \geq 1$. Hence, the lemma holds due to the following:

$$\begin{aligned}\Delta\Phi &= \sum_{e\in C'_i} c_e w_f^*(e) \log_2 w'_e \left(1 + \frac{1}{c_e}\right) - \sum_{e\in C'_i} c_e w_f^*(e) \log_2 w'_e \\ &= \sum_{e\in C'_i} c_e w_f^*(e) \log_2 \left(1 + \frac{1}{c_e}\right) \geq \sum_{e\in C'_i} w_f^*(e) \geq k_i - l.\end{aligned}$$

■

We have an upper bound on $\Delta c(w_f)$ as shown in the following lemma.

***Lemma** 2:* $\Delta c(w_f) < k_i - l$.

Proof: Let $\Delta w_f(e)$ be the increase of $w_f(e)$ in a single iteration of the weight increment. We then have the following:

$$\begin{aligned}\sum_{e\in C'_i} c_e \Delta w_f(e) &= \sum_{e\in C'_i} c_e \left(\min\left\{1, w'_e\left(1 + \frac{1}{c_e}\right)\right\} - w'_e\right) \\ &\leq \sum_{e\in C'_i} c_e \left(w'_e\left(1 + \frac{1}{c_e}\right) - w'_e\right) = \sum_{e\in C'_i} w'_e.\end{aligned}$$

Hence, to prove the lemma, we only need to show that $\sum_{e\in C'_i} w'_e < k_i - l$. We show this by contradiction. Assume the contrary, i.e., $\sum_{e\in C'_i} w'_e \geq k_i - l$. If $e \in C'_i$, then $w'_e < 1$ and thus $w_f(e) = w'_e$. Hence, we have $\sum_{e\in C'_i} w_f(e) \geq k_i - l$. This implies $\sum_{e\in C_i} w_f(e) \geq k_i$, since $C_i - C'_i$ contains l edges with $w_f(e) = 1$. However, this is a contradiction since the weight increment is performed only when $x(L_i; w_f) < k_i$, which implies that $\sum_{e\in C_i} w_f(e) < k_i$ because $x(L_i; w_f)$ is equal to the total weight of the edges in the r-L_i minimum-weight cut, C_i (by the max-flow min-cut theorem). Thus, the lemma follows. ■

Initially, the value of Φ is $-2\alpha_f \log_2 |E|$, and it never exceeds α_f since it always holds that $w'_e < 2$ for all $e \in E$. Hence, we have $2\alpha_f \log_2 |E| + \alpha_f$ as an upper bound on $\Delta\Phi$. Let m_j be the number of iterations of the weight increment when $k_i - l = j$, until the algorithm terminates. At each iteration of the weight increment when $k_i - l = j$, Φ increases by at least j (Lemma 1). Hence, the total increase of Φ until the algorithm terminates is at least $\sum_{\forall j} j \cdot m_j$. Since the total increase of Φ cannot exceed $2\alpha_f \log_2 |E| + \alpha_f$, it follows that $\sum_{\forall j} j \cdot m_j \leq 2\alpha_f \log_2 |E| + \alpha_f$. Also, at each iteration of the weight increment when $k_i - l = j$, $\Delta c(w_f)$ is at most j (Lemma 2). Hence, the cost $c(w_f)$ is bounded by:

$$c(w_f) \leq \sum_{\forall j} j \cdot m_j + 1 < 2\alpha_f \log_2 |E| + \alpha_f + 1. \quad (1)$$

Here, the additive term of one is due to the cost of the initial weight assignment. Finally, by multiplying the upper bound in Eq. (1) by 2, we can obtain $c(w_f) \leq 4\alpha_f \log_2 |E| + 2\alpha_f + 2$. Thus, Theorem 3 follows.

B. Competitive-Ratio Analysis of ROA-OMCND

In this subsection, we prove the following theorem.

***Theorem** 4:* ROA-OMCND achieves $O(\log |E| \log N')$ expected competitive ratio for OMCND, where N' is the minimum number of leaf nodes required to be connected to the root r to meet all demands arrived.

Let n_l be the minimum number of leaf nodes required to be connected to meet all previous demands up to d_l. Let E_{n_l} be the set of edges chosen by RR when demand d_l is given. Suppose that $d_i = (L_i, k_i)$ is the last demand. Then, $N' = n_i$. Let w_f be the fractional solution obtained by OAFS-OMCND. We have an upper bound on the average cost paid by RR as shown in the following lemma.

***Lemma** 3:* The expected cost of the edges chosen by RR is at most $O(\alpha_f \log |E| \log N')$.

Proof: The lemma holds due to the following:

$$\mathbb{E}\left[\sum_{e \in E_{N'}} c_e\right] \le \sum_{e\in E}\sum_{j=1}^{2\lceil \log_2(N'+1)\rceil} c_e \cdot \Pr\left[X(T_e, j) < w_f(e)\right]$$
$$\le \sum_{e\in E}\sum_{j=1}^{2\lceil \log_2(N'+1)\rceil} c_e w_f(e)$$
$$\le 2\lceil \log_2(N'+1)\rceil(4\alpha_f \log_2|E| + 2\alpha_f + 2).$$

The last inequality follows from the upper bound on w_f (Theorem 3). ■

We have an upper bound on the average cost paid by MCPDS as shown in the following lemma.

***Lemma** 4:* The expected cost of the edges chosen by MCDPS is at most $O(\alpha_f \log N')$.

Proof: Let $\mathcal{T}_{E_l}$ be the set of the subtrees that have no path of edges in E_l that connects a leaf node in L_l to the root r. Note that $x_q(L_l; w_f)$, i.e., the maximum flow from the root r to the leaf nodes of L_l on subtree T_q, must be equal to either the weight of the edge incident to the root of T_q, or the total weight of the edges incident to a leaf node of L_l in T_q. This implies that if $\theta(T_q) \ge x_q(L_l; w_f)$, then there would exist no path of edges in E_l that connects a leaf node of L_l in subtree T_q to the root r. Since the random variables $X(T_q, j)$, for a fixed j, are independent, the probability that MCDPS is invoked due to the random variables indexed by j is at most:

$$\Pr\left[X(T_q, j) \ge x_q(L_l; w_f) \text{ for all } T_q \in \mathcal{T}_{E_l}\right]$$
$$\le \prod_{T_q \in \mathcal{T}_{E_l}} (1 - x_q(L_l; w_f)) \le e^{-\sum_{T_q\in\mathcal{T}_{E_l}} x_q(L_l; w_f)} \le e^{-1}.$$

The last two inequalities follow, respectively, from the facts that $1 - x \le e^{-x}$ for $x \ge 0$, and that MCDPS is invoked only when $\sum_{T_q\in\mathcal{T}_{E_l}} x_q(L_l; w_f) \ge 1$. Since the random variables $X(T_q, j)$, for a fixed T_q and $1 \le j \le 2\lceil \log_2(n_l+1)\rceil$, are independent, the probability that MCDPS is invoked when d_l is given is at most:

$$\Pr\left[\theta(T_q) \ge x_q(L_i; w_f) \text{ for all } T_q \in \mathcal{T}_{E_n}\right] \le e^{-2\lceil \log_2(n_l+1)\rceil} < {n_l}^{-2}.$$

Recall that MCDPS chooses the path requiring the minimum additional cost. Obviously, the cost of the edges in E_a does not exceed α_f. Therefore, the expected cost paid by the single invocation of MCDPS is at most $\alpha_f {n_l}^{-2}$. Hence, the expected cost paid by MCDPS to meet all demands up to d_i is at most:

$$\alpha_f \sum_{l=1}^{i} {n_l}^{-2}(n_l - n_{l-1}) \le \alpha_f \sum_{l=1}^{N'} l^{-1} \le \alpha_f(\ln N' + 1),$$

where $n_0 = 0$. Thus, the lemma follows. ■

Due to Lemma 3, Lemma 4, and the fact that GPP adds no cost, we can obtain Theorem 4.

Finally, we have the following corollary from Theorem 4.

***Corollary** 1:* For MCMD-ID, ROA-OMCND can achieve $O\left(\log N \cdot (\log|Q| + \log \delta_{\max})\right)$ expected competitive ratio, where N is the minimum number of connections required to meet all redundancy requirements and $\delta_{\max}$ is the maximum inter-edge degree of any network node in Q. Further, provided that $\delta_{\max}$ is a constant, ROA-OMCND achieves $O(\log N \log|Q|)$ expected competitive ratio for MCMD-ID.

Proof: By the construction from MCMD-ID to OCMND, we have $N' = N$ and $|E| \le (2 + \delta_{\max})|Q|$. From this and Theorem 1 and Theorem 3, the corollary follows. ■

VI. Numerical Results

In this section, we evaluate the performance of ROA-OMCND through simulations. Ideally, we would like to compare the cost paid by ROA-OMCND to that of the optimal *offline* solution of MCMD-ID. However, since the offline problem of MCMD-ID is NP-hard (refer to [14]), we instead use its lower bound, denoted by LP-OPT, which we obtain by solving the LP relaxation of the offline problem of MCMD-ID.

The basic settings of the input parameters are as follows. We set $|P|$ and $|Q|$ to 100 and 200, respectively. We construct inter-edges between P and Q by randomly choosing d network nodes from Q as neighbors of each power node in P, where d is set to 5. We assign the costs $c_v(q)$ for $q \in Q$ (i.e., the cost of deploying a middleware gateway on q) and $c_l(p, q)$ for $p \in P$ and $q \in Q$ (i.e., the cost of establishing a communication link between p and q) by randomly picking an integer from $\{1, 2, \ldots, 5\}$, and then multiply c_l by ρ_c. Here, ρ_c is a factor for different ratios of $c_l(\cdot)$ to $c_v(\cdot)$, and is set to 0.01. At each run of the simulations, only a fraction of the remote devices, between 0.3 and 0.7, will be deployed one by one. At each time, one of the remote devices will be randomly selected as the one being deployed. We set the number of redundant connections to each remote device, i.e., r_H, to the same value for all $H \in \mathcal{H}$, in order to see the effect of the redundancy on the performance of the proposed algorithm. We perform simulations for different values of the parameters ρ_c, r_H, d, $|P|$ and $|Q|$.

Figure 3 shows the cost paid by ROA-OMCND for $r_H = 0, 1, 2$, as ρ_c varies from 0.001 to 10. We observe a noticeable gap between the cost paid by ROA-OMCND and the (estimated) offline minimum cost, when ρ_c has a small value, i.e., $\rho_c < 1$. The extra cost paid by ROA-OMCND, in addition to the offline minimum cost, goes up to 40% of the offline minimum cost. Recall that the offline minimum cost shown in the figure may be somewhat underestimated and further is not achievable in our setting, i.e., by any other online algorithm. Hence, note that the 40% additional cost (at maximum) *does not mean* that our proposed algorithm needs to pay 40% additional cost compared to the optimal/best online algorithm. On the other hand, we observe that the extra cost approaches to zero as ρ_c increases from 1. This is expected since the difficulty of MCMD-ID comes from different costs of deploying middleware gateways, i.e., $c_v(\cdot)$.

Figure 4 shows the cost paid by ROA-OMCND for $r_H = 0, 1, 2$, as d varies from 4 to 20. We observe that the offline minimum cost as well as the cost paid by ROA-OMCND decreases as d increases. We can understand this result by

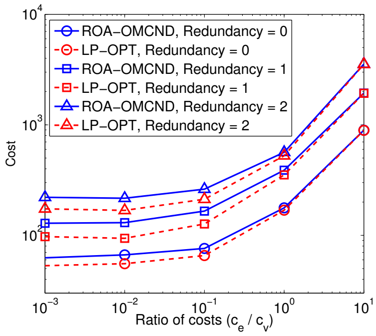


Fig. 3. Cost paid by ROA-OMCND for $r_H = 0, 1, 2$, as ρ_c varies from 0.001 to 10.

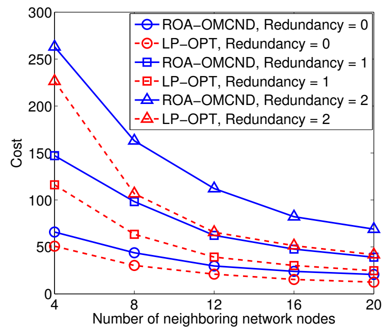


Fig. 4. Cost paid by ROA-OMCND for $r_H = 0, 1, 2$, as d varies from 4 to 20.

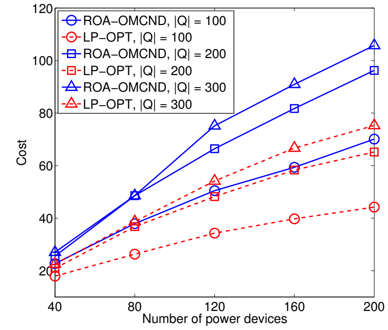


Fig. 5. Cost paid by ROA-OMCND for $|Q| = 100, 200, 300$, as $|P|$ varies from 40 to 200.

the following reasoning. As d increases, i.e., each remote device gets a larger number of neighboring network nodes, we would have more choices in deploying middleware gateways to provide the required connections. This would have perhaps reduced the middleware deployment cost. Also, we observe (in Fig. 3 as well) that the gap between the cost paid by ROA-OCMND and the offline minimum cost becomes larger as r_H increases. This result is likely to suggest that a higher redundancy makes problem more complicated, resulting in the gap being increased.

Figure 5 shows the cost paid by ROA-OMCND for $|Q| = 100, 200, 300$, as $|P|$ varies from 40 to 200. We observe that as $|P|$ grows, the gap between the cost paid by ROA-OCMND and the offline minimum cost increases, for all the values of $|Q|$. This result can be expected from our competitive-ratio analysis (see Corollary 1). That is, as $|P|$ grows, the number of neighboring remote devices of each network node increases and thus so does the maximum inter-edge degree of any network node. This increases the upper bound on the expected cost of ROA-OMCND, given in Corollary 1.

VII. Conclusion

In this paper, we proposed a middleware approach toward a robust and cost-effective smart grid communications system. Specifically, we studied the problem of how to design the middleware that facilitates the interplay between the power grid and the commercial communication infrastructure in a robust and cost-effective manner. We are particularly interested in the multi-stage deployment of power devices and their redundant connections. We formulated this problem into the minimum-cost middleware design under incremental deployment (MCMD-ID) problem. We designed a randomized online algorithm for MCMD-ID based on the reformulation of MCMD-ID into the online minimum-cost network design (OMCND). We proved that our proposed algorithm achieves the *order-optimal* expected competitive ratio, i.e., $O(\log N \log M)$ where N is the minimum number of connections required to meet all redundancy requirements and M is the number of commercial network nodes over which middleware is (potentially) deployed. Further, through simulations, we demonstrated that the proposed algorithm show reasonable performance, compared to the optimal offline solution.

We are currently designing a deterministic algorithm for MCMD-ID, e.g., by derandomizing the proposed randomized algorithm in this paper. Going forward, we plan to study attack-resilient middleware architecture design for secure smart grid communications.

References

[1] V. Terzija, G. Valverde, D. Cai, P. Regulski, V. Madani, J. Fitch, S. Skok, M. Begovic, and A. Phadke, "Wide-Area Monitoring, Protection, and Control of Future Electric Power Networks," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 80–93, January 2011.

[2] J. Y. Cai, Z. Huang, J. Hauer, and K. Martin, "Current Status and Experience of WAMS Implementation in North America," in *IEEE/PES Transmission and Distribution Conference and Exhibition: Asia and Pacific, Dalian, China*, 2005, pp. 1–7.

[3] C. H. Hauser, D. E. Bakken, and A. Bose, "A Failure to Communicate: Next Generation Communication Requirements, Technologies, and Architecture for the Electric Power Grid," *IEEE Power and Energy Magazine*, vol. 3, no. 2, pp. 47–55, March-April 2005.

[4] Y.-J. Kim, M. Thottan, V. Kolesnikov, and W. Lee, "A Secure Decentralized Data-Centric Information Infrastructure for Smart Grid," *IEEE Communications Magazine*, vol. 48, no. 11, pp. 58–65, November 2010.

[5] N. Kayastha, D. Niyato, E. Hossain, and Z. Han, "Smart Grid Sensor Data Collection, Communication, and Networking: A Tutorial," *Wireless Comm. and Mobile Computing, DOI: 10.1002/wcm.2258*, July 2012.

[6] H. Gjermundrod, D. E. Bakken, C. H. Hauser, and A. Bose, "GridStat: A Flexible QoS-Managed Data Dissemination Framework for the Power Grid," *IEEE Transactions on Power Delivery*, vol. 24, no. 1, pp. 136–143, January 2009.

[7] D. E. Bakken, A. Bose, C. H. Hauser, D. E. Whitehead, and G. C. Zweigle, "Smart Generation and Transmission With Coherent, Real-Time Data," *Proc. of the IEEE*, vol. 99, no. 6, pp. 928–951, June 2011.

[8] Y.-J. Kim, J. Lee, G. Atkinson, H. Kim, and M. Thottan, "SeDAX: A Scalable, Resilient, and Secure Platform for Smart Grid Communications," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 6, pp. 1119–1136, July 2012.

[9] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. S. Naor, "A General Approach to Online Network Optimization Problems," *ACM Transactions on Algorithms*, vol. 2, no. 4, pp. 640–660, October 2006.

[10] H. Qi, X. Wang, L. M. Tolbert, F. Li, F. Z. Peng, P. Ning, and M. Amin, "A Resilient Real-Time System Design for a Secure and Reconfigurable Power Grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 770–781, December 2011.

[11] N. Buchbinder and J. S. Naor, "The Design of Competitive Online Algorithms via a Primal–Dual Approach," *Foundations and Trends in Theoretical Computer Science*, vol. 3, no. 2-3, pp. 93–263, 2009.

[12] N. Alon, B. Awerbuch, and Y. Azar, "The Online Set Cover Problem," in *ACM STOC*, 2003.

[13] S. Korman, "On The Use of Randomization in The Online Set Cover Problem," M.S. Thesis, Weizmann Insititue of Science, 2005.

[14] D.-H. Shin, S. He, and J. Zhang, "Robust and Cost-Effective Architecture Design for Smart Grid Communications: A Multi-stage Middleware Deployment Approach," *Technical Report*, 2013. Available at http://www.public.asu.edu/~dshin17/pub/infocom14TR.pdf.