IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON GREEN CLOUD AND FOG COMPUTING: ENERGY EFFICIENT AND SUSTAINABLE INFRASTRUCTURES, PROTOCOLS AND APPLICATIONS

Received July 31, 2017, accepted September 8, 2017, date of publication September 21, 2017, date of current version October 25, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2755588

Battery Management in a Green Fog-Computing Node: a Reinforcement-Learning Approach

STEFANIA CONTI, GIUSEPPE FARACI, ROSARIO NICOLOSI, SANTI AGATINO RIZZO, AND GIOVANNI SCHEMBRA

Department of Electrical, Electronics and Computer Engineering, University of Catania, 95124 Catania, Italy

Corresponding author: Giovanni Schembra (schembra@dieei.unict.it)

This work was supported by the University of Catania through the Project on the Study on the interdependence of the electrical network and the Information and Communications Technology infrastructure in Smart Grid scenarios, FIR 2014.

ABSTRACT In the last years, Internet is evolving towards the cloud-computing paradigm complemented by fog-computing in order to distribute computing, storage, control, networking resources, and services close to end-user devices as much as possible, while sending heavy jobs to the remote cloud. When fog-computing nodes cannot be powered by the main electric grid, some environmental-friendly solutions, such as the use of solar- or wind-based generators could be adopted. Their relatively unpredictable power output makes it necessary to include an energy storage system in order to provide power, when a peak of work occurs during periods of low-power generation. An optimized management of such an energy storage system in a green fog-computing node is necessary in order to improve the system performance, allowing the system to cope with high job arrival peaks even during low-power generation periods. In this perspective, this paper adopts reinforcement learning to choose a server activation policy that ensures the minimum job loss probability. A case study is presented to show how the proposed system works, and an extensive performance analysis of a fog-computing node highlights the importance of optimizing battery management according to the size of the Renewable-Energy Generator system and the number of available servers.

INDEX TERMS Fog computing, renewable energy, battery management, reinforcement learning, Markov model.

I. INTRODUCTION

In the last years, we are assisting to a shifting process of the Internet towards the cloud-computing paradigm. As reported in Cisco Cloud Index (2013-2018), while since 2008 most Internet traffic was originated or terminated in a data center, in 2016 nearly two-thirds of total workloads in traditional IT space was processed in the cloud.

However, cloud-only models face serious challenges in latency, network bandwidth, geographic focus, reliability, and security [1]. With the goal of addressing this problem, fog computing emerged in the last years [2] as a new paradigm, which is not competing but complementary to cloud computing. More specifically, it is a horizontal architecture to distribute, when necessary, computing, storage, control, and networking resources and services close to end-user devices as much as possible, while sending heavy jobs to the remote cloud [3]–[5].

In the current literature, at the best of our knowledge, a common assumption is that fog-computing nodes are powered by energy coming from traditional electrical energy sources, which is always available whatever the requested amount of it. In that context, some works, highlighting the fog node peculiarity concerning the strict constraints on energy conservation, propose energy-efficient management schemes [6]–[9]. Other works limit their scope to propose energy consumption techniques specific for fog computing as compared with cloud computing [5]. Moreover, fog computing is combined to smart grids in [10] to provide energy management, and integrated with microgrids to support energy saving in IoT scenarios [11].

Nevertheless, in many application scenarios, fog-computing servers cannot be powered by the main electric grid, while traditional energy sources (e.g. diesel) involve environmental concerns. Therefore, the only possible power supply can be provided by renewable energy sources, e.g. solar or wind energy based generators. This is the case, for example, of wide rural and sparsely populated areas where computation is needed to elaborate data coming from sensors, cameras and drones covering it for different services [12], or desert, polar or marine areas to be monitored for

detection of dangerous situations, maintenance of integrity of production processes in large industrial plants in Industry 4.0 scenarios [13], [14], detection of fires or epidemic at plantations in smart farms [15], or even of emergency management scenarios in predictable (e.g., road accident, traffic jam) or unpredictable (e.g., earthquake, flooding) events [16].

The main problem is that the power produced by solar or wind generators is time-variable, strongly dependent on many factors like the area and the weather conditions where the sources are installed, the characteristics of the energy generators, and so on. Especially to cope with this random behavior of the power output from renewable generators, Battery Energy Storage Systems (BESS) have gained more and more importance in many applications of power system operation and management. In fact, the use of storage systems typically allows reducing or avoiding the problems caused by non-dispatchable distributed generators making their power output more continuous and somehow predictable. This is extremely important in isolated areas with high renewable energy penetration [17]. Electrical energy storage enables optimal power scheduling of renewable energy generators in both traditional distribution networks as well as in microgrids [18], economic benefits in power systems operation deriving from the optimal combination of generation, storage and loads [19], relief of congestions in transmission power systems [20], contribution of renewables to the ancillary market services [21], and so on.

Another intrinsic time-variant process in a fog-computing system is the process of job arrivals, generated by sources (e.g. sensors, cameras) that can be installed on either fixed (e.g. poles) or mobile positions (e.g., drones) [22]. Its behavior strongly depends on the applicative scenario, the kind of data and the way they are delivered to the fog servers (for example, in the case of drones, the arrival process may depend on the time-variant position of each drone transmitting data).

During periods when the arrivals exceed the number of servers or the power provided by renewable generator is not sufficient to supply all the needed servers, some jobs can be enqueued, or they are lost if the job queue saturates.

In order to face the variability of power availability, and avoid job queue saturation during periods of high job arrival rates, it is necessary to include a BESS. It can be recharged when the power provided by the generator exceeds the one needed to supply the servers. Nevertheless, management of the whole system, specifically the number of servers to be maintained active along the time, constitutes a challenging task to be optimally performed by a System Controller.

The target of this paper is to design a fog-computing node supplied by a renewable energy generator, where the SC optimally manages the BESS to minimize job loss probability. To this aim, a Markov-based analytical model of the system is integrated with a reinforcement learning process to optimize the server activation policy.

The paper is structured as follows. Section II describes the system, focusing on the renewable-energy based generator, the energy storage and the fog-computing data center components of the entire fog-computing node. Section III introduces two mathematical elements that play a key role in designing the system, that is, the switched batch Bernoulli process (SBBP) and the Reinforcement Learning (RL) process. Section IV illustrates the system model, while Section V mathematically derives the main performance parameters. Section VI applies the proposed system management technique to a case study and derives some numerical results. Finally, Section VII draws some conclusions.

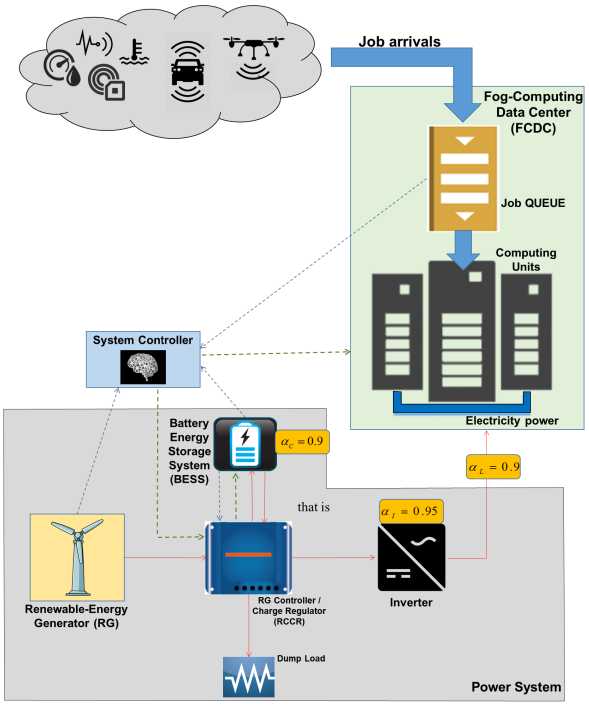


FIGURE 1. Reference System.

II. SYSTEM DESCRIPTION

The fog-computing node considered in this paper, whose architecture is sketched in Fig. 1, is made by three main parts: a *Fog Computing Data-Center* (FCDC), a *Renewable-Energy Generator* (RG) system, in the following assumed to be a wind generator without losing in generality, and a Battery Energy Storage System (BESS). The system is assumed to be off-grid, that is, it always works in autonomous mode of operation because it is not connected to the main power grid. Consequently, the BESS target is to cope time-variations of both the RG power output and the number of servers needed to provide computing facilities to arriving jobs.

The FCDC, constituted by N_S servers, has the objective of processing jobs that arrive according to a time-variant process whose statistics are known. Jobs that do not find an active server are enqueued in the job queue, in order to be processed later. Let us indicate the maximum queue size, that is, the maximum number of jobs that the job queue can contain, as Q_{MAX}. If the queue is full and there is no

sufficient power to supply an adequate number of servers to decrease the queue length when it is full, arriving jobs are rejected.

The behavior of the whole system is coordinated by a *System Controller* (SC), whose main target is to minimize the probability of job loss due to rejection for queue overflow. This task is performed by deciding how many servers in the FCDC maintaining active by means of the BESS when the RG power output is not sufficient. To this purpose, the SC takes into account the number of jobs waiting in the queue, the current state of the RG, the BESS state of charge (SOC) and the arrival process. This policy, as explained in the sequel, is optimized by means of a RL approach.

We assume that the SC performs its decisions periodically at each ΔT seconds. Accordingly, in the following we will characterize the whole system with discrete-time processes. The time variable, n, represents the current time slot, whose duration is equal to ΔT seconds.

Let us indicate the nominal power absorbed by each server as P_{Server}. Therefore, the maximum load at the input of the FCDC at the generic slot n depends on both the maximum number of servers that can be activated, and the current number, $S^{(Q)}(n)$, of jobs in the queue:

$$P_{Load}^{(MAX)}(n) = P_{Server} \cdot \min\left\{N_S,\ S^{(Q)}(n)\right\} \tag{1}$$

When the BESS is in *charging state*, the *RG Controller/Charge Regulator* (RCCR) block protects the BESS from overcharging, overload and overvoltage, besides managing the changes of the input voltage. On the other hand, when the BESS is in *discharging state*, the RCCR avoids "deep discharging" and adopts discharging strategies aiming at increasing battery life. The *Inverter* adapts the output voltage and frequency to the load requirements.

In order to evaluate the effective power and energy available to the FCDC system, it is necessary to take into account the efficiencies of the power system components. More specifically, α_I is the inverter efficiency, while α_C is the battery charge/discharge efficiency, whose typical values are, respectively, $\alpha_I = 0.95$ and $\alpha_C = 0.9$. Another coefficient to be considered is the power factor of the FCDC, α_L, assumed to be equal to 0.9.

The main effect of the presence of the inverter efficiency and the FCDC power factor is that a portion of the power generated by the RG, or supplied by the BESS, is unusable by the FCDC. Such a wasted power at the generic slot n when the power absorbed by the FCDC is $P_{Load}(n)$, is given by:

$$P_{WST}^{(FCDC)}(n) = (1 - \alpha_I\,\alpha_L)\,P_{Load}(n) \tag{2}$$

Consequently, in order to supply the maximum FCDC load at the generic slot n, it is necessary that the power system provides the following power:

$$P_{PS,OUT}^{(MAX)}(n) = \frac{P_{Load}^{(MAX)}(n)}{\alpha_I \alpha_L} \tag{3}$$

When the RG power output, $S^{(RG)}(n)$, is able to supply the maximum load, that is:

$$S^{(RG)}(n) \geq P_{PS,OUT}^{(MAX)}(n) \tag{4}$$

the BESS contribution is not required. Moreover, in this case, if the BESS is not completely charged, the RG will charge it with a power that depends on the BESS nominal power, $P_{Nom}^{(B)}$, assumed as the maximum power that can enter the BESS in one time slot, the current SOC, $S^{(SOC)}(n)$, expressed in terms of amount of energy in the BESS at the beginning of the slot n, and the residual power generated by the RG and not used by the load:

$$P_{BCh}(n) = \min\left\{\begin{matrix} P_{Nom}^{(B)},\ \left(B_{MAX} - S^{(SOC)}(n)\right)/\Delta T, \\ S^{(RG)}(n) - P_{PS,OUT}^{(MAX)}(n) \end{matrix}\right\} \tag{5}$$

where B_{MAX} is the maximum amount of energy that the BESS can store.

By assuming a simple linear charge/discharge behavior of the BESS, and defining δ_B as the number of slots needed to fully charge the BESS at nominal power $P_{Nom}^{(B)}$ when it is completely empty, the term B_{MAX} is given by:

$$B_{MAX} = (\delta_B \cdot \Delta T) \cdot P_{Nom}^{(B)} \tag{6}$$

During BESS charging periods, the SOC is modified as follows:

$$S^{(SOC)}(n+1) = S^{(SOC)}(n) + P_{BCh}(n)\,\Delta T \tag{7}$$

The amount of power generated by the RG that exceeds the power necessary to supply the maximum FCDC load and recharge the BESS is delivered to the dump load and lost in order to avoid RG damages. The corresponding wasted power at the slot n is given by:

$$P_{WST}^{(B)}(n) = S^{(RG)}(n) - P_{PS,OUT}^{(MAX)}(n) - P_{BCh}(n) \tag{8}$$

On the contrary, if the power generated by the RG is not sufficient to supply the maximum FCDC load at time slot n, i.e. if the condition (4) is not satisfied, additional servers that cannot be directly supplied by the RG could be supplied thanks to the BESS. In this case, the BESS is in discharging state, and the maximum power that it can provide in this condition is given by:

$$P_{BDech}^{(MAX)}(n) = \min\left\{\begin{matrix} P_{Nom}^{(B)},\ \alpha_C S^{(SOC)}(n)/\Delta T, \\ P_{PS,OUT}^{(MAX)}(n) - S^{(RG)}(n) \end{matrix}\right\} \tag{9}$$

Therefore, taking into account the charge/discharge efficiency of the BESS, the SOC during a discharging slot is modified as follows:

$$S^{(SOC)}(n+1) = S^{(SOC)}(n) - \frac{P_{BDech}(n)}{\alpha_C}\Delta T \tag{10}$$

where $P_{BDech}(n)$ is the BESS power output set by the SC at the slot n. In other words, according to the power that the BESS can supply, $P_{BDech}^{(MAX)}(n)$, the SC decides the number of additional servers to be activated to serve jobs that are waiting for service in the job queue. This is done through a policy that ensures the minimum job loss probability by a long-time point of view, as described later.

III. MATHEMATICAL PRELIMINARIES

In this section, we introduce two key elements needed to model the system described so far, that is, the SBBP model (Section III.A) and the reinforcement learning approach (Section III.B).

A. SBBP MODEL

A SBBP [23] is the most general Markov modulated process in the discrete-time domain. It is able to model a time-variant stochastic process whose behavior, described by a probability density function (pdf), is modulated by an underlying Markov chain. In this paper, we apply it to the job arrival process, representing the number of arrivals that occur in one slot.

According to the SBBP model definition in [24], an SBBP $\Lambda^{(X)}(n)$ can be characterized by the set $\left\{P^{(X)}, B^{(X)}, \Im^{(X)}, \Psi^{(X)}\right\}$, where:

- $P^{(X)}$ is the transition probability matrix of the underlying Markov chain of $\Lambda^{(X)}(n)$. If we describe this chain with the discrete-time process $S^{(X)}(n)$, the generic element of $P^{(X)}$ represents the transition probability from a state s_X to a state s'_X, that is:

$$P^{(X)}_{[s_X,\, s'_X]} = \Pr\left\{S^{(X)}(n+1) = s'_X \,\middle|\, S^{(X)}(n) = s_X\right\} \quad (11)$$

- $B^{(X)}$ is the arrival probability matrix describing the probability distribution of the number of arrivals of the process $\Lambda^{(X)}(n)$ for each state of the underlying Markov chain $S^{(X)}(n)$. Its generic element represents the probability that β jobs arrive in one slot when the state of the underlying Markov chain of $\Lambda^{(X)}(n)$ is s_X, that is:

$$B^{(X)}_{[s_X,\, \beta]} = \Pr\left\{\Lambda^{(X)}(n) = \beta \,\middle|\, S^{(X)}(n) = s_X\right\} \quad (12)$$

- $\Im^{(X)}$ is the state space of the underlying Markov chain $S^{(X)}(n)$;
- $\Psi^{(X)}$ is the set of possible values that the process $\Lambda^{(X)}(n)$ can assume, that is, the state space of the number of arrivals that can occur in one slot.

B. REINFORCEMENT LEARNING

The base of the RL problem is the interaction between an entity behaving as decision-maker, called *agent*, and a *system*, which is the environment where the agent operates. These two entities interact with each other continuously to achieve a given goal, consisting in maximizing over time some special system-specific numerical values, called rewards. In these interactions, the agent selects actions, and the system responds to those actions and presents new situations to the agent [25].

Four additional elements play a fundamental role in a RL problem: a *policy*, a *reward function*, a *state-value function* and the *system model* [25].

A *policy* defines the set of actions to be performed for all the system states. A *reward function* accounts for the goal of the agent. It assigns an immediate reward to an action performed in a given state, that is, a number indicating the intrinsic desirability of performing a given action when the environment is in a given state. The immediate reward also depends on the states that could be reached when this action is performed. On the converse, a *state-value function* assigns a quality measure to a state, by a long-term point of view. The value related to a given state accounts for the overall reward an agent could gather in the future when the system is in that state, hence highlighting its long-term goodness. Finally, the *system model* enables to account for potential states before they are actually experienced.

In this paper, the SC behaves as agent, while the action is represented by the number of additional servers that, according to a SC decision, are supplied by the BESS.

If the system state transitions do not depend on the previous history, but only on the current state and the performed action, then we say that the environment satisfies the Markov property. In this case, starting from the only knowledge of the current state, one can completely predict both the future behavior of the system and the respective expected rewards. A RL process that satisfies the Markov property is called Markov decision process (MDP). In addition, if both state and action spaces are finite, the model is called *finite MDP*. In this paper, we refer to this last kind of processes.

Let $\Im^{(\Sigma)}$ and $\Im^{(A)}$ be the sets of all the system states and all the possible actions the agent can perform, respectively. Let $S^{(\Sigma)}(n) \in \Im^{(\Sigma)}$ be the state of the system at the generic slot n, and $A(n) \in \Im^{(A)}$ the performed action at the same slot. Moreover, let Γ be the policy, that is, the set of 2-tuples (action, state), each representing the action $A(n)$ that the agent will perform when the system is in the state $S^{(\Sigma)}(n)$.

RL can be used in two different ways:

1) *Run-time mode*: during the learning process, at each slot the agent tries an action, and then it is reinforced by receiving an evaluation number, that is, the reward related to this action. In this case, the RL algorithm selects an action according to a given probability. More specifically, at each slot, say it n, the agent receives the current representation of the system state, $S^{(\Sigma)}(n)$, and, according to the policy Γ it is using, it decides an action $A(n) \in \Im^{(A)}$. At the next time slot, i.e. $n+1$, the agent receives both a numerical reward, $R(n+1)$, that is a consequence of the previous action, and the new system state, $S^{(\Sigma)}(n+1)$. To this purpose, it searches for the optimal policy by means of an online process: it updates the previously mentioned probabilities along the time in order to find the actions that maximize the received reward.
2) *Offline mode*: the optimal policy is found offline by solving a system of equations, called Bellman optimality equations, as explained below. In this way, the policy to decide actions for each state of the system is available to the agent since the beginning.

The first approach is used when there is no information on the system behavior (e.g. historical data are not available to model the RG power output and the job arrival process). The second approach, on the other hand, can be used when the

system, the model transition probabilities and the expected immediate rewards of the finite MDP are completely known. Moreover, when the optimal policy is found offline by means of the *Offline mode*, this policy can be used as starting point of the *Run-time mode*.

In this paper, we focus on the second approach because it assumed that the historical data are known. Therefore, in the sequel we will focus on how to find the optimal policy in a system where the transition probabilities are known.

In this paper, we apply the second approach. Therefore, in the sequel we will focus on how to find the optimal policy.

In order to characterize a finite MDP, for each action a and each starting state s_Σ at the slot n, let us define the *transition probability* towards the state s'_Σ at the slot $n+1$ and the *expected immediate reward* in the same slot $n+1$. These quantities, completely specifying the most important aspects of the dynamics of a finite MDP, are defined as follows:

$$p^{(\Sigma)}(s'_\Sigma \,|s_\Sigma,\, a) = \Pr\left\{ S^{(\Sigma)}(n+1) = s'_\Sigma \,\middle|\, S^{(\Sigma)}(n) = s_\Sigma,\, A(n+1) = a \right\} \tag{13}$$

$$r^{(\Sigma)}(s_\Sigma, s'_\Sigma, a) = \mathrm{E}\left\{ R(n+1) \,\middle|\, \begin{array}{l} S^{(\Sigma)}(n+1) = s'_\Sigma, S^{(\Sigma)}(n) = s_\Sigma, \\ A(n+1) = a \end{array} \right\} \tag{14}$$

where $\mathrm{E}\{\cdot\}$ is the expected-value operator, while $R(n+1)$ is the reward corresponding to the transition from the state s_Σ at the slot n to the state s'_Σ at the slot $n+1$, when the action a is performed.

For a given policy, Γ, the related *state-value function* associated to a given state s_Σ can be computed by summing the expected rewards along the time, after weighting them with a parameter $d \in [0, 1[$, referred to as discount-rate parameter [26]. The parameter d determines the present value of future rewards, and is used to weigh a reward received k slots in the future with a weight d^k. Of course, the smaller the value of d, the lower the weight given to future rewards. For example, after a huge number of numerical analysis of the system considered in this paper, we decided to use $d = 0.5$.

The *state-value function* associated to a given state s_Σ, representing the value of a state s_Σ under a policy Γ, is defined as follows:

$$v_\Gamma(s_\Sigma) = \mathrm{E}\left\{ \sum_{k=0}^{+\infty} d^k R(n+k+1) \,\middle|\, S^{(\Sigma)}(n) = s_\Sigma,\, \Gamma \right\} \tag{15}$$

The expression in (15) is called the Bellman equation for the state s_Σ.

Applying the theorem of total probability to consider all the actions a belonging to the policy Γ when the starting state is s_Σ, it is easy to demonstrate that the Bellman equation expressing the state-value function for the state s_Σ, can be recursively rewritten as follows:

$$v_\Gamma(s_\Sigma) = \sum_{a \in \Im^{(A)}_{s_\Sigma}} \Pr(a \,|s_\Sigma, \Gamma) \cdot \sum_{s'_\Sigma \in \Im^{(\Sigma)}} p^{(\Sigma)}(s'_\Sigma \,|s_\Sigma,\, a) \cdot \left[r^{(\Sigma)}(s_\Sigma, s'_\Sigma, a) + d\, v_\Gamma(s'_\Sigma) \right] \tag{16}$$

where $\Im^{(A)}_{s_\Sigma}$ represents the set of possible actions specified by Γ when the system is in the state s_Σ.The Bellman equation in (16) expresses a relationship between the value of a state and the values of its successive states.

Now, if we indicate the set of all possible policies that can be defined for the system as $\Im^{(\Gamma)}$, we say that a policy Φ is an *optimal policy* if its state-value function is better than or equal to the state-value function of all the other policies, for all the states of the system, that is:

$$v_\Phi(s_\Sigma) \geq v_\Gamma(s_\Sigma), \quad \forall s_\Sigma \in \Im^{(\Sigma)},\ \forall \Gamma \in \Im^{(\Gamma)} \tag{17}$$

Let us observe that:

- it is possible to demonstrate that a finite MDP admits at least one optimal policy [25];
- the system admits more than one optimal policy if more than one policy give the same optimal state-value function. Let us indicate the optimal state-value function shared by the set of optimal policies as $\tilde{v}(s_\Sigma)$.

Therefore, using the Bellman equation in (16), the *optimal state-value function* $\tilde{v}(s_\Sigma)$ can be written as follows [25]:

$$\tilde{v}(s_\Sigma) = \max_{a \in \Im^{(A)}_{s_\Sigma}} \left(\sum_{s'_\Sigma \in \Im^{(\Sigma)}} p^{(\Sigma)}(s'_\Sigma \,|s_\Sigma,\, a) \cdot \left[r^{(\Sigma)}(s_\Sigma, s'_\Sigma, a) + d\, \tilde{v}\left(s'_\Sigma\right) \right] \right) \tag{18}$$

The above equation is named the *Bellman optimality equation*.

Assuming that all the model transition probabilities and the expected immediate rewards of the finite MDP are completely known, we can now determine the set of optimal policies.

First of all, we will evaluate the optimal state-value function $\tilde{v}(s_\Sigma)$, for each $s_\Sigma \in \Im^{(A)}_{s_\Sigma}$. To this end, we write (18) for each state $s_\Sigma \in \Im^{(\Sigma)}$, obtaining a system of N_Σ nonlinear equations in N_Σ unknowns, where N_Σ is the cardinality of $\Im^{(\Sigma)}$.

Now we can easily find an optimal policy for the system, which aims at minimizing the job loss probability in the long-term. Recalling that, for each state $s_\Sigma \in \Im^{(\Sigma)}$, there are one or more actions giving the maximum in the Bellman optimality equation, it follows that whatever policy assigning a probability different from zero only to the actions that satisfy the expression in (18) is optimal.

In this paper, among the optimal policies available at each state, the SC chooses the one with the greatest number of

servers at work, in order to also maximize the short-term gain by minimizing the short-term job loss probability.

IV. SYSTEM MODEL

The time-variant behavior of the system described in the previous section can be captured by modeling the state of the SOC, the job arrival rate, the amount of power produced by the RG, and the state of the job queue. Therefore, we define the whole system state with the following 4-dimension discrete-time Markov chain:

$$S^{(\Sigma)}(n) = \left(S^{(SOC)}(n), S^{(A)}(n), S^{(Q)}(n), S^{(RG)}(n)\right) \quad (19)$$

where:

- $S^{(SOC)}(n)$ represents the SOC. Its state space, $\Im^{(SOC)} = \{b_1, \ldots, b_G\}$, is constituted by a number G of quantized levels of charge, where b_G is the maximum amount of energy that can be stored in the BESS, that is, $b_G = B_{MAX}$;
- $S^{(A)}(n)$ is the state of the underlying Markov chain of the SBBP $\Lambda^{(A)}(n)$ modeling the job arrival rate. As said in the previous section, it is completely described by the set $\left\{p^{(A)}, B^{(A)}, \Im^{(A)}, \Psi^{(A)}\right\}$;
- $S^{(Q)}(n)$ is the state of the job queue, i.e. the number of jobs that are waiting for a service in the job queue. If Q_{MAX} indicates the maximum number of jobs that the queue can contain, its state space is $\Im^{(Q)} = \{0, \ldots, Q_{MAX}\}$;
- $S^{(RG)}(n)$ represents the amount of power generated by the RG in the slot n, as defined in Section II. Let $\Im^{(RG)} = \{g_1, \ldots, g_W\}$ be its state space, constituted by W levels of generation power. Of course, the state g_W coincides with the RG nominal power, $P^{(RG)}_{Nom}$.

In order to derive the transition probability matrix of the considered system, we assume the following sequence of events in the transition from the slot n to the slot $n+1$, as shown in Fig. 2:

1) System departures: jobs which are running on the servers leave them after having received the service during the previous slot;
2) SC decision: the SC, observing the state of the system at the beginning of the slot $n+1$, decides the number a of servers (besides the ones that can be supplied by the power directly coming from the RG) to be supplied by means of the BESS during the slot $n+1$;
3) Dequeue: a number of jobs, equal to the number of servers at work during the slot $n+1$, leave the queue and enter the servers;
4) Job arrivals: new jobs arrive with a probability distribution determined by the state of the underlying Markov chain $S^{(A)}(n)$, according to the SBBP modeling the job arrival process;
5) Update of the BESS charge state: the state variable $S^{(SOC)}(n)$, representing the SOC in the slot n, is updated as in (7) or (10), depending on whether the condition in (4) is true or false;

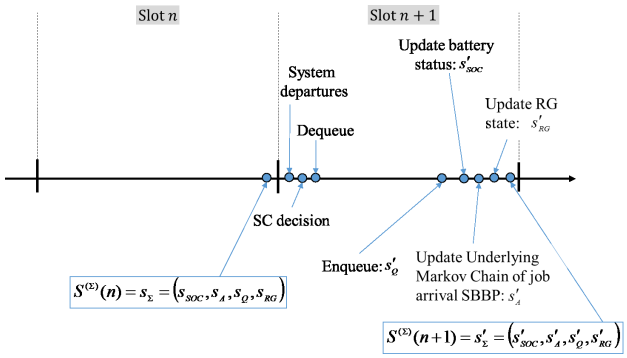


FIGURE 2. Sequence of event in each slot.

6) Update of the underlying Markov chain $S^{(A)}(n)$ of the job arrival SBBP process $\Lambda^{(A)}(n)$;
7) Update of the state $S^{(RG)}(n)$ of the power generation Markov chain;
8) System state observation: the state $S^{(\Sigma)}(n+1)$ is observed with all the modifications due to the previous events in the slot $n+1$.

The total number of servers that work in the generic slot n is $F = k + a$, where k is the number of servers supplied by the RG according to the current state s_{RG}, while a is the number of servers supplied by the BESS according to the decision taken by the SC following the policy Γ calculated off-line by the RL algorithm. F determines (and represents) the number of job departures from the queue.

The number k can be easily derived as follows:

$$k = \left\lfloor \frac{s_{RG}\,\alpha_I \alpha_L}{P_{Server}} \right\rfloor \quad (20)$$

where $\lfloor x \rfloor$ represents the maximum integer contained in x.

Now, let us define the transition probability matrix of the whole system. To this purpose, let us consider two generic states of the system, $s_\Sigma = (s_{RG}, s_A, s_Q, s_{SOC})$ and $s'_\Sigma = (s'_{RG}, s'_A, s'_Q, s'_{SOC})$, representing the start and the arrival states of a generic transition from the slot n to the slot $n+1$.

Taking into account the definition of the states s_Σ and s'_Σ, and applying the theorem of the total probability, we can rewrite (13), which represents the generic element of the transition probability matrix $p^{(\Sigma)}(a)$ of the state of the whole system for a given action a, as follows:

$$p^{(\Sigma)}_{[s_\Sigma, s'_\Sigma]}(a) = p^{(\Sigma)}(s'_\Sigma \,|s_\Sigma,\, a) = p^{(RG)}_{[s_{RG}, s'_{RG}]} \cdot p^{(A)}_{[s_A, s'_A]} \cdot p^{(Q)}_{[s_Q, s'_Q]}(s_A, s_{RG}, a) \cdot p^{(SOC)}_{[s_{SOC}, s'_{SOC}]}(s_Q, s_{RG}, a) \quad (21)$$

where:

- $p^{(RG)}_{[s_{RG}, s'_{RG}]}$ is the generic element of the transition probability matrix of the Markov chain modeling the RG power output process, known as input of the problem;

- $p^{(A)}_{[s_A,s'_A]}$ is the generic element of the transition probability matrix of the underlying Markov chain of the job arrival SBBP, known as input of the problem;
- $p^{(Q)}_{[s_Q,s'_Q]}(s_A, s_{RG}, a)$ is the generic element of the transition probability matrix of the queue state. This transition depends on the state s_A of the underlying Markov chain of the job arrival process, the RG state, and the number a of servers to be supplied by the BESS.
- $p^{(SOC)}_{[s_{SOC},s'_{SOC}]}\left(s_Q, s_{RG}, a\right)$ is the generic element of the transition probability matrix of the SOC. It depends on both the state s_{RG} of the RG and the number a of servers supplied by the BESS.

The term $p^{(Q)}_{[s_Q,s'_Q]}(s_A, s_{RG}, a)$ can be calculated by applying the total probability theorem on the number of job arrivals in one slot, i.e. for each $\beta \in \Psi^{(A)}$, and accounting that, in a transition starting from the state s_Q, the arrival queue state s'_Q is equal to the starting state plus the number of arrivals, truncated to the maximum queue size Q_{MAX}, if necessary, minus the number of departures. Therefore, we have:

$$p^{(Q)}_{[s_Q,s'_Q]}(s_A, s_{RG}, a) = \sum_{\forall\beta\in\Psi^{(A)}} B^{(A)}_{[s_A,\beta]} \cdot I\left(s_Q, s'_Q, s_{RG}, \beta, a\right) \tag{22}$$

where $I\left(s_Q, s'_Q, s_{RG}, \beta, a\right)$ is a Boolean indicator function of the feasibility of state transition from s_Q to s'_Q, defined as follows:

$$I\left(s_Q, s'_Q, s_{RG}, \beta, a\right) = \begin{cases} 1 \text{ if } s'_Q = \min\left\{s_Q - F + \beta, Q_{MAX}\right\} \\ 0 \qquad\qquad \text{otherwise} \end{cases} \tag{23}$$

In order to evaluate $p^{(SOC)}_{[s_{SOC},s'_{SOC}]}\left(s_Q, s_{RG}, a\right)$, first let us evaluate the new SOC state, s'_{SOC}, as discussed in Section II. We distinguish between two different cases.

If the BESS is in *charging state*, i.e. $s_{RG} \geq P^{(MAX)}_{PS,OUT}$, where $P^{(MAX)}_{PS,OUT}$ can be derived from (3) and (1) by setting $S^{(Q)}(n) = s_Q$, according to (7) we have:

$$soc' = s_{SOC} + P_{Bch} \cdot \Delta T \tag{24}$$

where P_{Bch} can be calculated as in (5) by considering that the SOC state and the RG state at the slot n are s_{SOC} and s_{RG}, respectively, while $P^{(MAX)}_{PS,OUT}$ can be expressed in terms of the job queue state s_Q as said so far. Therefore, we have:

$$P_{Bch} = \min\left\{\begin{matrix} P^{(B)}_{Nom}, \ (B_{MAX} - s_{SOC})/\Delta T, \\ s_{RG} - P^{(MAX)}_{PS,OUT} \end{matrix}\right\} \tag{25}$$

Otherwise, if the BESS is in *discharging state*, i.e. $s_{RG} < P^{(MAX)}_{PS,OUT}$ and the SC sets the number a of servers supplied by the BESS greater than zero, we can derive s'_{SOC} according to (10), that is:

$$soc' = s_{SOC} - \frac{P_{Bdech}}{\alpha_C}\Delta T \tag{26}$$

where P_{Bdech} can be calculated as:

$$P_{Bdech} = a\frac{P_{server}}{\alpha_I\alpha_L} - \left(s_{RG} - \frac{k\,P_{server}}{\alpha_I\alpha_L}\right) \tag{27}$$

The term in parenthesis accounts for the residual power from RG, which is not sufficient for supplying an additional server without the BESS. Let us notice that the quantity soc' calculated as in (24) or (26) for the two cases of BESS in charging and discharging state, may be a value not belonging to the set $\Im^{(SOC)}$. For this reason, to calculate the transition probability element $p^{(SOC)}_{[s_{SOC},s'_{SOC}]}(s_{RG}, a)$, where s'_{SOC} is one of the G quantized levels of $\Im^{(SOC)}$, we will assume that the actual SOC arrival state is one of the two closest values of $\Im^{(SOC)}$ to soc' (i.e. the highest value that is lower than soc', indicated as s'_{SOC_L}, or the lowest value that is higher than soc', indicated as s'_{SOC_H}). The choice of the specific arrival state of the transition between the two above values is assumed with a probability that depends on the distances of soc' from them. Therefore, the generic element of the SOC transition probability matrix can be calculated as follows:

$$p^{(SOC)}_{[s_{SOC},s'_{SOC}]}\left(s_Q, s_{RG}, a\right) = \begin{cases} 1 - \frac{\left|s'_{SOC_H} - soc'\right|}{\left|s'_{SOC_H} - s'_{SOC_L}\right|} & \text{if } soc' \leq b_G \text{ and } s'_{SOC} = s'_{SOC_H} \\ 1 - \frac{\left|soc' - s'_{SOC_L}\right|}{\left|s'_{SOC_H} - s'_{SOC_L}\right|} & \text{if } soc' > b_1 \text{ and } s'_{SOC} = s'_{SOC_L} \\ 1 & \text{if } soc' < b_1 \text{ and } s'_{SOC} = b_1 \\ 1 & \text{if } soc' > b_G \text{ and } s'_{SOC} = b_G \\ 0 & \text{otherwise} \end{cases} \tag{28}$$

where:

$$s'_{SOC_H} = \min\left(b_i \in \Im^{(SOC)} : b_i \geq soc'\right)$$
$$s'_{SOC_L} = \max\left(b_i \in \Im^{(SOC)} : b_i < soc'\right) \tag{29}$$

Now, we have all the matrices needed to build the matrix $p^{(\Sigma)}(a)$ whose generic element, for each action a, was defined in (21). In order to apply the RL process to decide the optimum policy Γ, we still need to calculate the expected reward $r^{(\Sigma)}(s_\Sigma, s'_\Sigma, a)$ associated to the transition from the state s_Σ to the state s'_Σ, for each action a , as defined in (14). To this purpose, we have defined the reward as a function of the job loss probability. More specifically, we define the reward corresponding to the transition from the state s_Σ at the slot n to the state s'_Σ at the slot $n+1$, when the action a is performed as the number of the jobs that are lost, changed in sign to be a reward and not a penalty, that is:

$$R(n+1) = -Loss(n+1) \tag{30}$$

Therefore, the best reward is zero, and the reward worsens as the number of lost job increases.

The expected reward associated to the transition from the state s_Σ to the state s'_Σ, following the policy Γ, can be

computed according to (14) as follows:

$$r^{(\Sigma)}(s_\Sigma, s'_\Sigma, a) = -\mathrm{E}\left\{ Loss(n+1) \left| \begin{matrix} S^{(\Sigma)}(n+1) = s'_\Sigma,\ S^{(\Sigma)}(n) = s_\Sigma \\ A(n+1) = a \end{matrix} \right. \right\} \tag{31}$$

In order to derive the number of jobs that are lost during the transition from the state s_Σ to the state s'_Σ, let us take into account that, according to the sequence of events illustrated in Fig. 2, losses may occur only if the queue arrival state is $s'_Q = Q_{MAX}$. Such a condition is necessary but not sufficient to cause a job loss. More specifically, if the queue starting state is s_Q, β new jobs arrive to the FCDC system, and $F = k+a$ servers are working, the queue can accommodate at most $Q_{MAX} - s_Q + F$ jobs. Therefore, some job losses occur if β is greater than this value. Specifically, the number of jobs that are lost because no space is available in the queue for them is $\beta - (Q_{MAX} - s_Q + F)$.

Thus, the expected value in (30) can be calculated as follows:

$$\mathrm{E}\left\{ Loss(n+1) \left| \begin{matrix} S^{(\Sigma)}(n+1) = s'_\Sigma,\ S^{(\Sigma)}(n) = s_\Sigma \\ A(n+1) = a \end{matrix} \right. \right\} = \begin{cases} \sum_{\beta = Q_{MAX} - s_Q + F + 1}^{\beta_{MAX}} \left[\beta - Q_{MAX} + s_Q - F\right] \cdot B^{(A)}_{[s_A, \beta]} \\ \quad \text{if } s'_Q = Q_{MAX} \\ 0 \\ \quad \text{otherwise} \end{cases} \tag{32}$$

where $B^{(A)}_{[s_A,\beta]}$ is the element $[s_A, \beta]$ of the job arrival probability matrix, representing the probability that β jobs arrive when the underlying Markov chain of the SBBP $\Lambda^{(A)}(n)$ is s_A.

Now, we have all the elements to apply the reinforcement learning in offline mode to calculate the optimum policy Γ, as described in Section III-B. Let us observe that the feasible range for the number of servers that can be activated when the system is in the state s_Σ is a subset of $\Im^{(A)}_{s_\Sigma} = \{0, ..., a_{MAX}\}$, where a_{MAX} can be calculated accounting for both the maximum power that can be provided by the BESS, and the residual RG power which is not sufficient to supply a further server, that is:

$$a_{MAX} = \left\lfloor \frac{\alpha_I \alpha_L \left[P^{(MAX)}_{B_D ech} + \left(s_{RG} - k \frac{P_{Server}}{\alpha_I \alpha_L} \right) \right]}{P_{Server}} \right\rfloor \tag{33}$$

$\lfloor x \rfloor$ representing the maximum integer contained in x. The term $P^{(MAX)}_{B_D ech}$ can be derived from (9) as follows:

$$P^{(MAX)}_{B_D ech} = \min \left\{ \begin{matrix} P^{(B)}_{Nom},\ \alpha_C s_{SOC} / \Delta T, \\ P^{(MAX)}_{PS,OUT} - s_{RG} \end{matrix} \right\} \tag{34}$$

Finally, once the RL has been applied, the resulting optimum policy Φ gives us the best action a to be performed for each transition from the state s_Σ to the state s'_Σ. By substituting the values of a in (21) for each 2-tuple s_Σ and s'_Σ, we obtain the overall transition probability matrix of the system $\tilde{p}^{(\Sigma)}$.

We can now derive the steady-state probability array, $\pi^{(\Sigma)}$, for this system when the System Controller applies the optimum policy Φ, whose generic element is:

$$\pi^{(\Sigma)}_{[s_\Sigma]} = \mathrm{Prob}\left\{ S^{(\Sigma)}(n) = s_\Sigma \,|\text{Optimal policy } \Phi \right\} \tag{35}$$

It can be calculated, as known, by solving the following linear equation system:

$$\begin{cases} \pi^{(\Sigma)} \cdot \tilde{p}^{(\Sigma)} = \pi^{(\Sigma)} \\ \sum_{\forall s_\Sigma \in \Im^{(\Sigma)}} \pi^{(\Sigma)}_{[s_\Sigma]} = 1 \end{cases} \tag{36}$$

V. PERFORMANCE EVALUATION

Applying the model described in the previous section, now we derive the main performance parameters characterizing the behavior of the system.

The main parameter is the job loss probability, since the goal of the RL application by the SC regards the minimization of the per-slot number of job losses. It can be calculated as the ratio between the mean number of lost jobs in a slot and the mean number of arrived job in a slot, that is:

$$P_{Loss} = \frac{E\{Loss\}}{E\left\{\Lambda^{(A)}(n)\right\}} \tag{37}$$

where the numerator can be calculated as in (31), that is:

$$E\{Loss\} = \sum_{s_{SOC}=b_1}^{b_L} \sum_{s_A \in \Im^{(A)}} \sum_{s_{RG} \in \Im^{(RG)}} \sum_{\beta \in \Psi^{(A)}} \sum_{s_Q = Q_{MAX} - \beta}^{Q_{MAX}} \left(s_Q + \beta - Q_{MAX}\right) \cdot B^{(\Sigma)}_{[s_A, \beta]} \cdot \pi^{(\Sigma)}_{[s_{RG}, s_A, s_Q, s_{SOC}]} \tag{38}$$

while the denominator can easily be derived by the matrices characterizing the job arrival SBBP $\Lambda^{(A)}(n)$:

$$E\left\{\Lambda^{(A)}(n)\right\} = \sum_{s_A \in \Im^{(A)}} \sum_{\beta \in \Psi^{(A)}} \beta \cdot B_{[s_A, \beta]} \cdot \pi^{(A)}_{[s_A]} \tag{39}$$

Another important parameter that characterizes the performance of the considered system is the mean value of the delay suffered by the jobs in the queueing system, usually referred to as mean response time. It can be easily derived by means of the Little theorem [27], as follows:

$$E\{T\} = \frac{E\left\{N^{(\Sigma)}_{Jobs}\right\}}{E\left\{\Lambda^{(A)}(n)\right\}} \tag{40}$$

where the numerator can be derived by the steady-state probability array calculated in (35) as follows:

$$E\left\{N^{(\Sigma)}_{Jobs}\right\} = \sum_{s_{SOC}=b_1}^{b_L} \sum_{s_A \in \Im^{(A)}} \sum_{s_{RG} \in \Im^{(RG)}} \sum_{s_Q=0}^{Q_{MAX}} s_Q \cdot \pi^{(\Sigma)}_{[s_{RG}, s_A, s_Q, s_{SOC}]} \tag{41}$$

Now, let us observe that, at the planning stage, a tradeoff between costs and system performance is necessary. In this perspective, increasing the size of RG and BESS reduces the job loss probability against higher costs. However, on the other hand, inexpensive solutions could let to poor system performance. Optimal economic planning needs to consider the costs of specific components, but this kind of analysis is out of the scope of this paper. Notwithstanding, some related general insight can be pointed out by using the following indicator giving information about the average wasted power with respect to the nominal power of the RG. Such a quantity also depends on the size of RG with respect to the other system components.

$$I_{RG} = \frac{E\left\{P_{WST}^{(B)}(n)\right\}}{P_{Nom}^{(RG)}} \tag{42}$$

where $P_{Nom}^{(RG)}$ is the RG nominal power, while $E\left\{P_{WST}^{(B)}(n)\right\}$ is the mean wasted power. This last term can be derived by averaging the wasted power defined in (8) and calculated for each system states $s_\Sigma = \left(s_{RG}, s_A, s_Q, s_{SOC}\right)$, with the respective steady-state probability derived from the solution of the linear equation system in (35). We have:

$$E\left\{P_{WST}^{(B)}(n)\right\} = \sum_{s_\Sigma \in \Im^{(\Sigma)}} P_{WST\ [s_\Sigma]}^{(B)} \cdot \pi_{[s_\Sigma]}^{(\Sigma)} \tag{43}$$

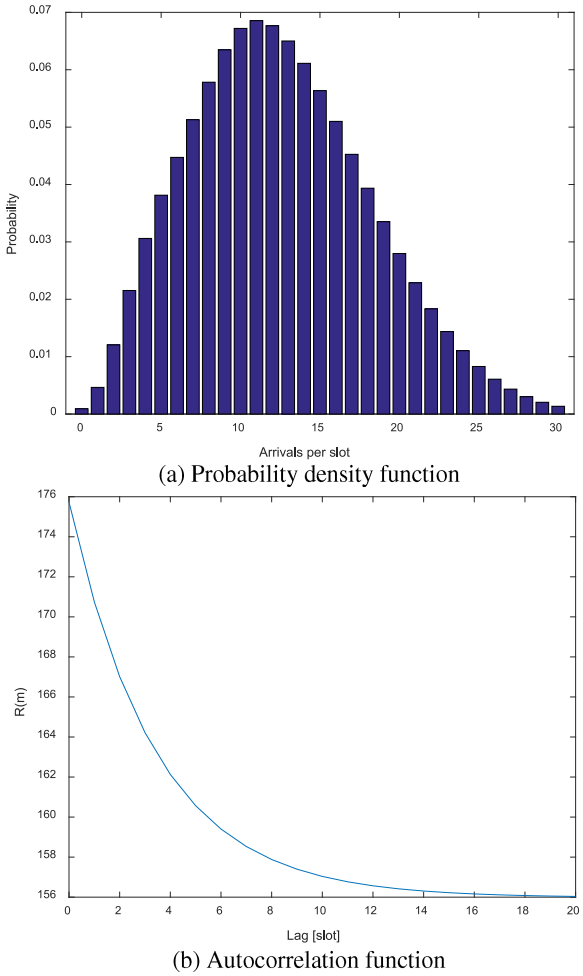
According to its definition, the indicator ranges in the interval [0,1]. When the value is close to 1, the RG could be oversized, causing power waste. On the other hand, for low values of I_{RG}, the RG size could be insufficient to supply the system. Actually, the value strongly depends also on the RG's average power output. More specifically, a value close to zero could be obtained due to a too small average power output with respect to the load. Therefore, without other information, the indicator value could let to misleading results. In this perspective, when it is coupled with the value of job loss probability, the indicator effectively supports the designer in choosing the best tradeoff for the sizes of RG and BESS. More specifically, for a given performance target, i.e. for an assigned job loss probability, the solutions with minimum I_{RG} are the most efficient and probably (it depends on the specific components' cost) the most expensive ones.

VI. CASE STUDY AND NUMERICAL RESULTS

In this section, we apply the BESS management policy proposed in this paper to a case study, which will be described in Section VI.A. Numerical results will be presented in Section VI.B.

A. SYSTEM SETUP

We consider a fog-computing node where the SC applies the proposed RL-based management policy every $\Delta T = 15$ min. The system is loaded by a job arrival process characterized by the pdf and the autocorrelation function shown in Fig. 3. They were measured on the field, while generated by motion-detection cameras deployed on a countryside populated by



(a) Probability density function

(b) Autocorrelation function

FIGURE 3. Statistics of the job arrival process.

birds and other small terrestrial animals. By applying the discrete-time inverse eigenvalue technique described in [28], from the above functions we derived a 4-state SBBP characterized by the following transition probability matrix:

$$P^{(A)} = \begin{bmatrix} 0.754 & 0.203 & 0.032 & 0.011 \\ 0.049 & 0.746 & 0.157 & 0.048 \\ 0.032 & 0.164 & 0.759 & 0.045 \\ 0.017 & 0.054 & 0.187 & 0.742 \end{bmatrix} \tag{44}$$

The rows of the job-arrival probability matrix are depicted in Fig. 4.

As far as the FCDC is concerned, we considered a set of INTEL Nuke MiniPCs, chosen for their reduced weight and dimension, each with an Intel®Core™i7-7567U Processor,4M Cache, 4.00 GHz, RAM of 16 Gbyte, and characterized by a power consumption of $P_{Server} = 65$ W. Their number, N_S, was varied in the range [12], [36] to evaluate its impact on the overall performance. The maximum size of the Job Queue was fixed to $Q_{MAX} = 60$.

The values of the efficiency coefficients of the power system, i.e. α_C, α_I and α_L, are shown in Fig. 1.

The RG is a wind generator, installed on the same countryside area. In order to calculate the power generation model,

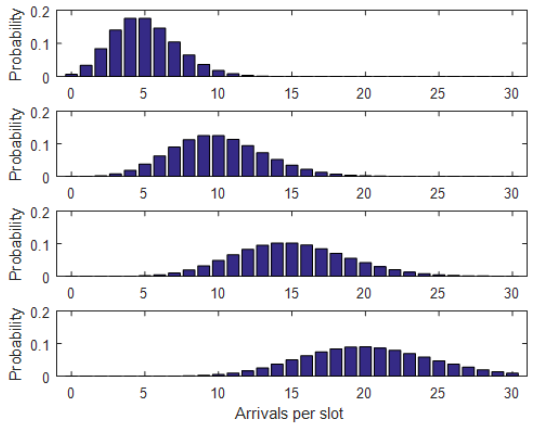


FIGURE 4. Job arrival probability matrix.

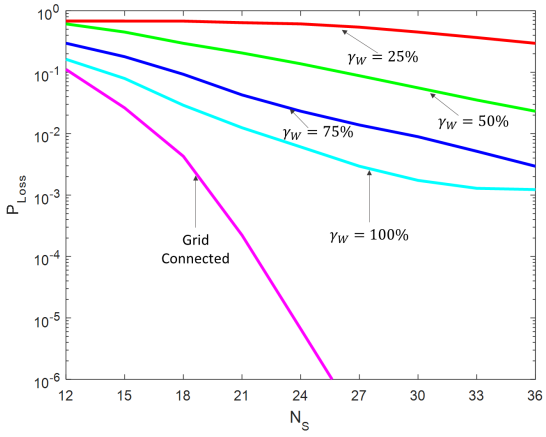


FIGURE 6. Job loss probability without BESS.

we measured its output at each 15 minutes in the period ranging between 02/20/2015 and 05/20/2017. The samples measured in the first 48 hours of this period are plotted in Fig. 5. Let us notice that the RG never was out of service for fault during the whole measurement period, this occurrence has been not considered. On the other hand, the case of lack of RG supply due to wind absence has been considered. The mean value of the RG power output is $\bar{P}_{RG} = 1520$ W. Therefore, taking into account the inverter efficiency, $\alpha_I = 0.95$, and the FCDC power factor, $\alpha_L = 0.9$, and that each server requires a power of $P_{Server} = 65$ W, the considered RG system is able to supply an average number of $\bar{P}_{RG}\,\alpha_I\alpha_L / P_{Server} = 20$ servers. The power generation model was derived by quantizing the trace with levels of 152 W, equivalent to the power needed to supply two servers.

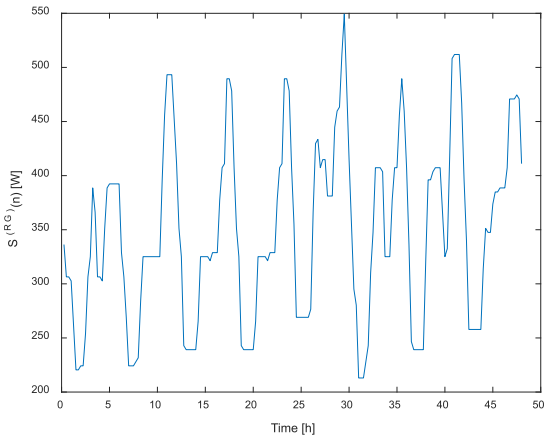


FIGURE 5. Trace of the power generated by the wind in two days.

In the following, in order to analyze cases with a number of servers, N_S, different from 20, and to supply them with a RG power generator that is able to provide a percentage, γ_W, of the power required by them, we derive the RG power generation model by starting from a rescaled version of the trace, calculated from the original one as follows:

$$\hat{w}(n) = \gamma_W \frac{N_S}{20} w(n) \tag{45}$$

The model is then obtained, as illustrated so far, by quantizing it with the same quantization step of 152 W.

Finally, as far as the BESS is concerned, we will consider a BESS whose nominal power depends on the chosen number of servers as follows:

$$P_{Nom}^{(B)} = \gamma_B \frac{N_S \cdot P_{Server}}{\alpha_I \cdot \alpha_L} \tag{46}$$

where γ_B is a coefficient that will be varied in the next section to analyze the impact of the BESS capabilities in supplying the active servers. Moreover, the SOC is quantized in 20 levels, evenly spaced in the interval $[0, B_{MAX}]$, with B_{MAX} is calculated as in (6), with $\delta_B = 4$ slots, i.e. one hour.

B. NUMERICAL RESULTS

In this section, we will present some numerical results achieved by applying the analytical model to design the optimum policy for the SC, in such a way that it can decide the number of server to be supplied by the BESS with the final goal of minimizing job loss probability.

Numerical results have been obtained by using a value iteration algorithm implemented in Matlab by the same authors. Although not extremely optimized, thanks to the Matlab computing toolbox it is able to obtain each point of the curves shown in the sequel in about 13 minutes by using a DELL PowerEdge R630 server with dual processor Intel Xeon E5-2630L and 384 GB of RAM. It is an acceptable time for off-line derivations.

Fig. 6 shows the job loss probability for increasing number of servers in absence of BESS and, consequently, without any use of the RL. The lowest curve shows results in case of grid-connected system, where the job loss probability only depends on the number of servers. This case has been introduced as a reference case because it presents the best hypotetical values the system could achieve when it operates in off-grid mode. In fact, the other curves, presenting

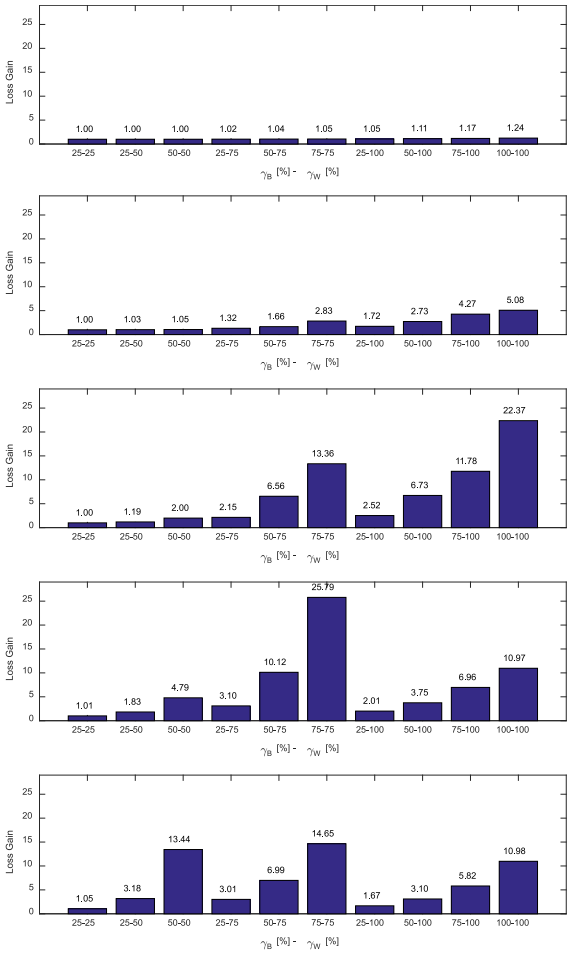


FIGURE 7. Performance improvements achieved by using the BESS.

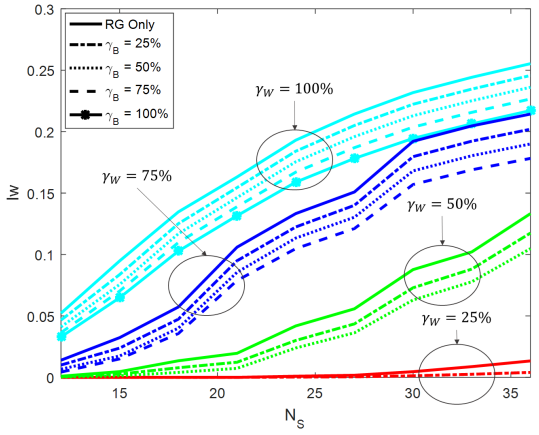


FIGURE 8. Wasted power indicator.

results for the system operating in off-grid mode, report worse performance, whose degradation depends on the RG size. More specifically, variuos scenarios are accounted by considering different values for the coefficient γ_W, whose effects are shown in (45). For example, the curve labeled with $\gamma_W = 75\%$ reports results obtained in a scenario where the mean RG power output is the 75% of the power needed to supply N_S servers. Consequently the RG in case of $N_S = 15$ in the curve $\gamma_W = 100\%$ is the same of the one obtained for the case $N_S = 30$ in the curve $\gamma_W = 50\%$. Nevertheless, the latter outperforms the former thanks to the greater number of servers although supplied by the same RG.

In order to evaluate performance improvement achieved thanks to the presence of BESS, optimally managed by the SC according to the RL application, we define a loss gain parameter as the ratio between the loss probabilities without and with BESS. Its values are shown in Fig. 7 for different 2-tuples of BESS and RG nominal powers. More specifically, the first term refers to the value of γ_B, representing the capacity of the BESS to supply the maximum number of available servers, while the second term is γ_W, representing the wind generator size, already presented above.

When the number of servers is equal to the average number of arrivals (i.e. $N_S = 12$), the addition of a BESS has no effect on the system performance, or lead to a little improvement. In this scenario, condition (4) is rarely satisfied, especially when the RG size is small, then the BESS is recharged few times and, consequently, it can be infrequently used. Therefore, the maximum number of servers that are supplied with the BESS differs from zero rarely, and then RL is infrequently applied, thus its effect is negligible.

Instead, as the number of servers increases, condition (4) is more frequently satisfied, representing the case that the RG power output is greater than the maximum power absorbed by the load. Since the RG size increases proportianally with the number of servers, the only reason leading to a more frequent occurence of condition (4) satisfaction is the increment of the probability of a job queue smaller than the number of servers. The increment in the number of times the BESS is recharged leads to more cases where the RL can be adopted, and provides it with a great number of choices to opt, that is increases the maximum number of servers that can be supplied by the BESS.

When the number of servers is greater than the maximum number of arrivals, which is 30 as shown in Fig. 3.a, the probability that the queque is smaller than the number of servers increases and then condition (4) is very frequently satisfied. This implies that the number of times the BESS is in discharge state is reduced, then the performance impovements achieved thanks to the use of the BESS an the application of the RL are reduced. For example, the last subplot of Fig. 7 shows that the RL in case of a 2-tuple (BESS, RG) given by $\gamma_B = 75\%$ enables a greater performance improvement than the case where the same BESS with a larger RG size is considered. The same effect is highligheted in the fourth subplot of Fig. 7, i.e. the one obtained for $N_S = 30$.

In order to analyze the amount of wasted power, in Fig. 8 we plotted the values of the index I_W, defined as in (39). The curves confirm that condition (4) is satisfied more frequently as the number of servers incrases, and the BESS reduces the wasted power especially for large RG size.

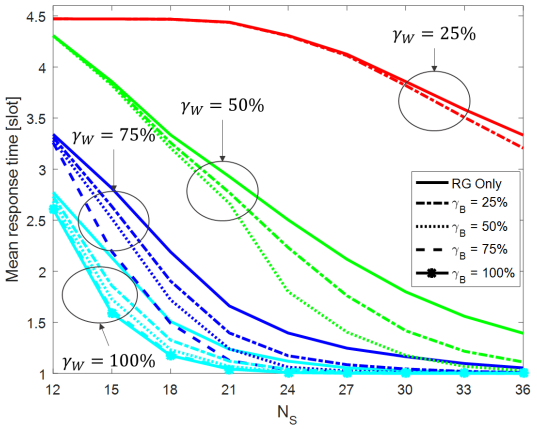


FIGURE 9. Mean response time of the FCDC node.

Finally, Fig. 9 shows the mean response time of the overall FCDC node, representing how much time a job spends in the node to be served. Of course, we can minimize this parameter by increasing the number of servers, by improving the RG system (i.e. by increasing the γ_W parameter). Moreover, the figure confirms that performance is further increased by using a BESS, and shows the importance of its design especially in cases when the RG is well sized (see the group of curves at the bottom of the figure).

VII. CONCLUSIONS

A common assumption in the current literature, at the best of our knowledge, is that fog-computing nodes are powered by energy coming from traditional electrical energy sources, which is always available whatever the requested amount of it. Nevertheless, in many application scenarios, fog-computing servers can be powered only by renewable energy sources.

In order to face the variability of power availability with this kinds of generators, and avoid job queue saturation during periods of high job arrival rates, this paper aims at designing a fog-computing node supplied by a renewable energy generator, where the SC optimally manages the BESS to minimize job loss probability. A Markov-based analytical model of the system is integrated with a reinforcement learning process to optimize the server activation policy.

A case study is presented to show how the proposed system works. An extensive performance analysis of a fog-computing node highlights the importance of optimizing battery management according to the size of the Renewable-Energy Generator system and the number of available servers.

REFERENCES

[1] B. McMillin and T. Zhang, "Fog computing for smart living," *Computer*, vol. 50, no. 2, p. 5, Feb. 2017.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. ACM MCC*, Helsinki, Finland, Aug. 2012, pp. 13–16.

[3] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for Internet of Things and analytics," in *Big Data and Internet of Things: A Roadmap for Smart Environments*, N. Bessis and C. Dobre, Eds. Cham, Switzerland: Springer, 2014.

[4] J. D. Glover, M. S. Sarma, and T. J. Overbye, *Power System Analysis and Design*, 5th ed. Boston, MA, USA: Cengage Learning, 2011.

[5] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1728–1739, May 2016.

[6] G. Faraci and G. Schembra, "An analytical model to design and manage a green SDN/NFV CPE node," *IEEE Trans. Netw. Service Manage.*, vol. 12, no. 3, pp. 435–450, Sep. 2015.

[7] G. Faraci and G. Schembra, "An analytical model for electricity-price-aware resource allocation in virtualized data centers," in *Proc. IEEE ICC*, London, U.K., Jun. 2015, pp. 5839–5845.

[8] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2016.2551747.

[9] S. Wang, X. Huang, Y. Liu, and R. Yu, "CachinMobile: An energy-efficient users caching scheme for fog computing," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Chengdu, China, Jul. 2016, pp. 1–6.

[10] M. A. Al Faruque and K. Vatanparvar, "Energy management-as-a-service over fog computing platform," *IEEE Internet Things J.*, vol. 3, no. 2, pp. 161–169, Apr. 2016.

[11] F. Jalali, A. Vishwanath, J. de Hoog, and F. Suits, "Interconnecting Fog computing and microgrids for greening IoT," in *Proc. IEEE Innov. Smart Grid Technol.-Asia (ISGT-Asia)*, Melbourne, VIC, Australia, Nov./Dec. 2016, pp. 693–698.

[12] C. Rametta and G. Schembra, "Designing a softwarized network deployed on a fleet of drones for rural zone monitoring," *Future Internet J.*, vol. 9, no. 1, p. 8, Mar. 2017.

[13] K. Zhou, T. Liu, and L. Zhou, "Industry 4.0: Towards future industrial opportunities and challenges," in *Proc. 12th Int. Conf. Fuzzy Syst. Knowl. Discovery (FSKD)*, Zhangjiajie, China, Aug. 2015, pp. 2147–2152.

[14] F. Shrouf, J. Ordieres, and G. Miragliotta, "Smart factories in Industry 4.0: A review of the concept and of energy management approached in production based on the Internet of Things paradigm," in *Proc. IEEE Int. Conf. Ind. Eng. Eng. Manage.*, Bandar Sunway, Malaysia, Dec. 2014, pp. 697–701.

[15] M. C. Dwarkani, R. R. Ganesh, S. Jagannathan, and R. Priyatharshini, "Smart farming system using sensors for agricultural task automation," in *Proc. IEEE Technol. Innov. ICT Agricult. Rural Develop. (TIAR)*, Chennai, India, Jul. 2015, pp. 49–53.

[16] H. Cruickshank, E. Bovim, A. Donner, J. Sesena, and R. Mort, "Reference scenarios for the deployment of emergency communications for earthquakes and mass transport accidents," in *Proc. 7th Adv. Satellite Multimedia Syst. Conf., 13th Signal Process. Space Commun. Workshop (ASMS/SPSC)*, Livorno, Italy, Sep. 2014, pp. 262–268.

[17] P. Yang and A. Nehorai, "Joint optimization of hybrid energy storage and generation capacity with renewable energy," *IEEE Trans. Smart Grid*, vol. 5, no. 4, pp. 1566–1574, Jul. 2014.

[18] S. Singh, M. Singh, and S. C. Kaushik, "Optimal power scheduling of renewable energy systems in microgrids using distributed energy storage system," *IET Renew. Power Generat.*, vol. 10, no. 9, pp. 1328–1339, Oct. 2016.

[19] V. Kalkhambkar, R. Kumar, and R. Bhakar, "Joint optimal allocation methodology for renewable distributed generation and energy storage for economic benefits," *IET Renew. Power Generat.*, vol. 10, no. 9, pp. 1422–1429, 2016.

[20] H. Khani, M. R. D. Zadeh, and A. H. Hajimiragha, "Transmission congestion relief using privately owned large-scale energy storage systems in a competitive electricity market," *IEEE Trans. Power Syst.*, vol. 31, no. 2, pp. 1449–1458, Mar. 2016.

[21] P. Zou, Q. Chen, Q. Xia, G. He, and C. Kang, "Evaluating the contribution of energy storages to support large-scale renewable generation in joint energy and ancillary service markets," *IEEE Trans. Sustain. Energy*, vol. 7, no. 2, pp. 808–818, Apr. 2016.

[22] C. Yu, J. Wang, J. Shan, and M. Xin, "Multi-UAV UWA video surveillance system," in *Proc. 14th Int. Conf. Control, Autom., Robot. Vis. (ICARCV)*, Phuket, Thailand, Nov. 2016, pp. 1–6.

[23] O. Hashida, Y. Takahashi, and S. Shimogawa, "Switched bath Bernoulli process (SBBP) and the discrete-time SBBP/G/1 queue with application to statistical multiplexer performance," *IEEE J. Sel. Areas Commun.*, vol. 9, no. 3, pp. 394–401, Apr. 1991.

[24] A. Lombardo, G. Morabito, and G. Schembra, "Modeling intramedia and intermedia relationships in multimedia network analysis through multiple timescale statistics," *IEEE Trans. Multimedia*, vol. 6, no. 1, pp. 142–157, Feb. 2004.
[25] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2012.
[26] V. François-Lavet, R. Fonteneau, and D. Ernst, "How to discount deep reinforcement learning: Towards new dynamic strategies," in *Proc. NIPS Workshop Deep Reinforcement Learn.*, Montreal, QC, Canada, Dec. 2015, pp. 7–12.
[27] A. Lombardo and G. Schembra, "Performance evaluation of an adaptive-rate MPEG encoder matching IntServ traffic constraints," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 47–65, Feb. 2003.
[28] J. D. C. Little, "A proof for the queuing formula: L = λW," *Oper. Res.*, vol. 9, no. 3, pp. 383–387, 1961.

STEFANIA CONTI received the degree (Hons.) in 1997 and the Ph.D. degree in electrical engineering from the University of Catania, Italy, in 2001. In 2002, she joined the Department of Electrical, Electronics and Computer Engineering with the University of Catania (UniCT), where she is currently an Associate Professor of electric power distribution, utilization and smart grids. Since 2012, she has been responsible for the activities of the EnSiEL Section in Catania (EnSiEL is a consortium of Italian Universities operating for research and dissemination in the area of Electrical Power Systems), established at DIEEI-UniCT. Her research interests include power systems reliability, protection and control; integration of distributed generation with distribution networks and smart grids; autonomous and non-autonomous operation of micro-grids; optimization techniques applied to electrical power systems.

Since 2015, she is a member of the Governing Council of EnSiEL as a representative of UniCT. She is a member of AEIT since 2002, the IEEE Power and Energy Society since 1997 and the IEEE Industrial Electronics Society since 2010.

GIUSEPPE FARACI was born in Syracuse, Italy, in 1989. He received the degree in telecommunication engineering in 2014. He was with the University of Catania in the project Sinergreen to define analytical models of smart grids. He currently collaborates with the National Inter-University Consortium for Telecommunications for the EU project INPUT.

His main research interests include green networking, smart grids, cloud computing, 5G networks, software defined networks and network function virtualization.

ROSARIO NICOLOSI received the degree in electrical engineering in 2001 and the Ph.D. degree in electrical engineering in 2012 from the University of Catania, Italy. He is currently a Contract Researcher with the Department of Electrical, Electronics and Computer Engineering with the University of Catania.

His main research interests concern power systems analysis and management, stand-alone and grid-connected distributed generators, analysis and modeling of renewable generation systems, interaction between distributed generators and distribution networks, and distribution system reliability.

SANTI AGATINO RIZZO received the degree (Hons.) in electronics engineering in 2004 and 2006, respectively, and the Ph.D. in electrical engineering from the University of Catania, Italy, in 2010. He is currently a Research Fellow with the Department of Electrical, Electronics, and Computer Engineering, University of Catania.

His research interests include distribution networks reliability analysis; distributed generation; renewable energies; wireless power transfer; application of numerical methods, stochastic optimization, and machine learning in the field of electrical engineering.

GIOVANNI SCHEMBRA received the degree (Hons.) in 1991, and the Ph.D. degree in electronics, informatics and telecommunications engineering in 1995. He is currently an Associate Professor with the University of Catania. From 1991 to 1992, he was with the Telecommunications Research Group of the Cefriel of Milan, involved in traffic modeling and performance evaluation in broadband networks. He was involved in several national and EU projects. In particular, he was involved for the University of Catania in the European Project DOLMEN (Service Machine Development for an Open Long-term Mobile and Fixed Network Environment), and has been acting as WP Leader in the NoE Newcom. He is involved for CNIT to the H2020 INPUT (In-Network Programmability for next-generation personal cloUd service supporT) project. His research interests mainly concern with software defined networks, network functions virtualization, 5G networks, traffic modeling, cloud computing and data center management, mobile cloud networks.