Approximate Dynamic Programming for Stochastic Resource Allocation Problems

Ali Forootani, *Member, IEEE*, Raffaele Iervolino, *Member, IEEE*, Massimo Tipaldi, and Joshua Neilson

***Abstract*—A stochastic resource allocation model, based on the principles of Markov decision processes (MDPs), is proposed in this paper. In particular, a general-purpose framework is developed, which takes into account resource requests for both instant and future needs. The considered framework can handle two types of reservations (i.e., specified and unspecified time interval reservation requests), and implement an overbooking business strategy to further increase business revenues. The resulting dynamic pricing problems can be regarded as sequential decision-making problems under uncertainty, which is solved by means of stochastic dynamic programming (DP) based algorithms. In this regard, Bellman's backward principle of optimality is exploited in order to provide all the implementation mechanisms for the proposed reservation pricing algorithm. The curse of dimensionality, as the inevitable issue of the DP both for instant resource requests and future resource reservations, occurs. In particular, an approximate dynamic programming (ADP) technique based on linear function approximations is applied to solve such scalability issues. Several examples are provided to show the effectiveness of the proposed approach.**

***Index Terms*—Approximate dynamic programming (ADP), dynamic programming (DP), Markov decision processes (MDPs), resource allocation problem.**

I. Introduction

RESOURCE allocation is defined as the set of problems in which one has to assign resources to tasks over some finite time horizon to customer requests. Many important real-world matters can be cast as resource allocation problems, including applications in air traffic flow management [1], energy [2], logistics, transportation, and fulfillment [3]. These problems are notoriously difficult to solve for two reasons. First, they typically exhibit stochasticity, i.e., the requests to be processed may arrive randomly according to some stochastic process which, itself, depends on where resources are allocated. Second, they exhibit extremely large state and action spaces, making solution by traditional methods infeasible [4], [5]. In the fields of operational research and artificial intelligence, primarily, the state space as well as decisions (or actions) are discrete. In this regard, resource allocation problems with discrete states and decisions are studied at length under the umbrella of Markov decision processes (MDPs) [6]. Systems with uncertainty and nondeterminism can be naturally modelled as MDPs [7], [8]. For instance, emotion recognition in text [9], speaker detection [10], and fault-tolerant routing [11] are considered as MDPs.

The optimal policy for MDPs can be computed by applying exact dynamic programming (DP) techniques thanks to their strength in solving sequential decision making problems [7]. However, it is well known that such techniques suffer from the *Curse of Dimensionality*, which is due to state and action space explosion of real-world applications [8]. For this reason, efforts have been devoted to finding the techniques able to solve this problem in an approximate way [12]. This field has evolved under a variety of names including approximate dynamic programming (ADP), neuro-dynamic programming, and reinforcement learning [13]–[15].

In this paper, resource allocation problems are formulated and solved via a general-purpose MDP-based framework, which can be used for different real business contexts. We address both instant (i.e., customers requires a resource to be allocated immediately) and advance (i.e., the customer books a resource for future use) resource requests. It is considered that the same resource can be sold at different price values at different times to take the advantage of heterogeneous preferences of customers over time, e.g., a seat on an airplane or a room in a hotel. Both the formulation and the corresponding resolution for resource allocation problems with instant resource requests were firstly explored by the authors in [16], where only exact DP approaches were applied. In [17], the authors further extended the approach to incorporate the possibility that, besides an immediate allocation request, a customer can book a resource in advance for future utilisation. Two types of booking procedures were considered, that is to say, booking resources with specified and unspecified time interval options.

The main differences of this paper with [17] are the following: i) new assumptions and procedures both in modeling and in the resource reservation approach; ii) the usage of the unweighted sample based least squares (USLS)-ADP algorithm instead of a temporal difference ADP based approach [18] to solve the curse of dimensionality; iii) comprehensive and analytical algorithms suitable for

Manuscript received March 12, 2020; accepted April 13, 2020. Recommended by Associate Editor Qinglai Wei. *(Corresponding author: Ali Forootani.)*

Citation: A. Forootani, R. Iervolino, M. Tipaldi, and J. Neilson, "Approximate dynamic programming for stochastic resource allocation problems," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 4, pp. 975–990, Jul. 2020.

A. Forootani is with the Hamilton Institute, Maynooth University, Co. Kildare W23 F2K8, Ireland (e-mail: Aliforootani@gmail.com).

R. Iervolino is with the Department of Electrical Engineering and Information Technology, University of Naples, Napoli 80125, Italy (e-mail: rafierv@unina.it).

M. Tipaldi and J. Neilson are with the Department of Engineering, University of Sannio, Benevento 82100, Italy (e-mail: mtipaldi@unisannio.it; neilson@unisannio.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2020.1003231

computer based implementation. As for the first aspect, the proposed solution manages the overbooking situations, which occur when the number of allocated resources at the current time slot can not be confirmed since new resources have been already allocated for the next one (due to the advance resource reservation mechanism), and the overall needs can not be satisfied by the system overall capacity. Such overbooking situations occur since the system handles both instant and future resource requests. Managing them entails a significant update in the modelling of resource allocation problems, its dynamics, and the resulting allocation policy.

The USLS-ADP algorithm was presented for the first time by the authors in [19], where its convergence properties over an infinite time horizon are also discussed. The USLS-ADP algorithm inherits both the contraction mapping and monotonicity properties of the exact discounted DP algorithms [20]. Thanks to this, it is suited for both finite and infinite time horizons. As a consequence, it can be used for solving resource allocation problems with instant and advance reservation requests. The latter case actually involves time intervals over finite (possibly very large) time horizons.

As for the third aspect, we provide the implementation mechanisms of the reservation pricing algorithms, starting from the steps defined in [17]. For instance, we show how to exploit Bellman's principle of optimality [7] to assess the allocation of future resources at different prices and their impact on the current expected total revenue. More specifically, a stochastic prediction of the system evolution (up to the time when the new resource is requested) is performed. Then, the set of possible prices is applied and assessed based on how they affect the current expected total revenue. When the most suitable price is chosen, the complete pricing policy is renewed. This approach shows how to bridge the gap between model predictive control (MPC) and DP [13].

Various parts of the proposed modeling and optimization approach, consisting of DP, reservation procedure, and ADP, are implemented in the MATLAB environment for resource allocation problems in a general framework. Moreover, different examples are provided to support and evaluate the effectiveness of the method.

This paper is organized as follows. Section II shows how to model resource allocation problems via MDPs. Resource allocation problem modeling with specified time interval reservation requests and the related pricing algorithm are provided in Sections III–V. Resource allocation problems with unspecified reservation time intervals are outlined in Section VI. Section VII addresses the usage of the proposed reservation pricing algorithm for resource allocation problems with large state space. Simulation results are provided in Section VIII. Section IX outlines the scientific literature relevant to this work. Finally, Section X concludes the paper.

II. Preliminaries and Modelling Resource Allocation Problems as MDPs

This section shows how to formulate resource allocation problems as a set of constrained parallel discrete-time birth death processes (BDPs) [21], which are integrated into one Markov decision process (MDP). The configuration of the resulting MDP based framework can be controlled by the price manager, who assigns a price among m possible choices by applying a specific pricing policy.

Such approach was firstly introduced by the authors in [16]. Hereafter, the main aspects of such framework are outlined along with the notation used in this manuscript. We also provide some preliminaries on how to solve the related decision making problem via DP based techniques [7], [20]. For more details, the reader can refer to [16].

A. MDP Notation

The following MDP notation is adopted in the paper:

- $\Xi = \{\xi^1, \ldots, \xi^\Omega\}$ is the finite set of states, where $\xi^v, \xi^w \in \Xi$ denote two generic elements of this set, with $v, w = 1, \ldots, \Omega$. The state variable at the generic time slot j is denoted with $\xi(j) \in \Xi$. Note that, for the sake of generality, a symbolic notation for the MDP states is used.
- $U = \{u_1, \ldots, u_n\}$ is the finite set of actions (also called decisions or controls), where u is a generic element of this set. We define the control function $u(\xi, j)$ as the mapping between the whole state space Ξ and the set of actions U, at the time slot j. For the sake of simplicity, in the paper we remove the explicit dependency on the state, i.e., $u(\xi, j) := u(j)$.
- $p_{\xi^v \xi^w}(u) := [p(\xi(j+1) = \xi^w | \xi(j) = \xi^v, u(j) = u)]$ is the state transition probability function. It gives the probability that an action $u \in U$, performed in the state ξ^v at the time slot j, leads the system to the state ξ^w at the time slot $j+1$.
- $R : \Xi \times U \to [0, +\infty)$ is the reward function, obtained when taking an action u_i at any generic state $\xi^v \in \Xi$. In this paper, we consider that reward function only depends on the state, so it can be written as $R(\xi^v)$.

B. Modeling Resource Allocation Problems as MDP

This paragraph addresses the problem of dynamically pricing $N \in \mathbb{N}$ equivalent resources and allocating them to customers. A set of m hourly prices (or prices per unit of time) is given, and price managers can select one of them in order to maximize the expected total revenue. They can also reject resource requests from customers, if deemed not convenient from a profit standpoint. Price managers can charge different prices for the same resource over time depending on the resource availability and expected profit. As shown in [16], it is possible to formulate such price management system as a set of BDPs. In particular, there is a dedicated BDP for each feasible price c_i (with $i = 1 \ldots m$), which allows modelling the unpredictable behavior of customers in requesting and releasing resources. As a result, the system evolves as a set of parallel BDPs. By assigning a specific price at each time slot, the price manager defines which BDP is active for one (possible) birth and one (possible) death, whereas all the others are active only for one (possible) death. In this regard, we assume that:

- At maximum, only one customer can request a resource at each time slot. Moreover, each customer can request it for either immediate or future reservation (the latter addressed in the next sections).
- The time slot duration is chosen so that, for each price c_i, at most one customer associated to each BDP may leave at

any time slot.

This way, we can establish the decision making process by integrating all the BDPs into one MDP.

Having said that, we can proceed by providing the mathematical formulation of resource allocation problems for a specific price. The MDP *associated with each price* c_i is defined by a tuple $C_i = \langle \Xi_i, U_i, \mathcal{T}_i, R_i \rangle$ where,

• Ξ_i is the state space, $\Xi_i = \{\xi_i^0, \xi_i^1, \ldots, \xi_i^N\}$. The state variable at time j is denoted with $\xi_i(j) \in \Xi_i$. Moreover, we denote with $\xi_i^{v_i}$ and $\xi_i^{w_i}$ two generic states of the process. In simpler words, $\xi_i(j) = \xi_i^{v_i}$ means that v_i resources are allocated at time j at price c_i.

• The maximum number of allocable resources is assumed to be finite and equal to N, and we define $|\xi_i^{v_i}| = v_i \leq N$.

• U_i is the finite set of actions (also called decisions or controls), defined as $U_i = \{c_i, \nu\}$, where $c_i \in \mathbb{R}_+$ represents "allocation" with price c_i and ν denotes the action of "rejection". With a slight abuse of notation, we denote with $U_i(\xi_i)$ the set of actions admissible for the state variable $\xi_i(j)$. Hence, for each state ξ_i, we have

$$\begin{cases} U_i(\xi_i) = \{c_i, \nu\}, & \text{if } |\xi_i| < N \\ U_i(\xi_i) = \{\nu\}, & \text{if } |\xi_i| = N. \end{cases} \tag{1}$$

The decision at time j is denoted as $u_i(j) \in U_i(\xi_i(j))$. As previously defined, $N \in \mathbb{N}$ is the number of resources.

• $\mathcal{T}_i$ is the state transition mapping, represented by the state transition probability matrix with elements

$$\begin{aligned} p_{\xi_i^{v_i}\xi_i^{w_i}}(u_i(j)) &:= p[\xi_i(j+1) = \xi_i^{w_i} | \xi_i(j) = \xi_i^{v_i}, u_i(j)] \\ &\text{s.t. } \max\{0, v_i - 1\} \leq w_i \leq \min\{N, v_i + 1\} \end{aligned} \tag{2}$$

where the constraint implies the fact that the system is modelled as a BDP. Therefore, all the pairs of states not fulfilling it have associated transition probabilities equal to zero.

• $R_i : \Xi_i \to [0, +\infty)$ is the reward function. In our case

$$R_i(\xi_i) = c_i \xi_i \tag{3}$$

where, for the sake of simplicity and with a slight abuse of notation, the state variable ξ_i is used instead of the corresponding allocated number of resources v_i.

The MDP C_i shown in Fig. 1 depicts the state transition probabilities among the various states. By applying the decision $u_i = c_i$, the resulting Markov chain C_i allows a birth transition from the current state with probability λ_i (representing the probability that a customer requires the resource at price c_i), a death transition with probability μ_i (representing the probability a customer releases a resource previously reserved at price c_i), and a self-transition with probability $1 - \lambda_i - \mu_i$ (representing the probability that no customer releases or asks for a resource). On the other hand, when the decision $u_i = \nu$ is taken, no customer can purchase the resource, and only a death transition or a self-transition from the current state is allowed.

The "composition" of m different C_i corresponding to each price and a common constraint representing the fact that the number of available resources is equal to N, give rise to the overall system C. In particular, it can be modelled as a

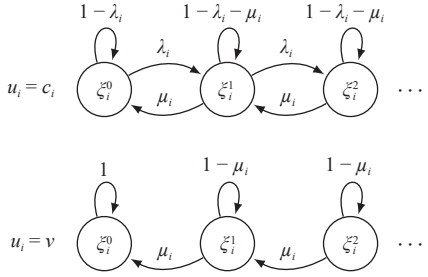

Fig. 1. MDP C_i. Transitions allowed with the input $u_i = c_i$ are depicted in the top part of the figure. Transitions allowed with the input $u_i = \nu$ are depicted in the bottom part.

constrained time-homogeneous (or stationary) MDP, defined by a tuple $C = \langle \Xi, U, \mathcal{T}, R \rangle$ where

• Ξ is the entire state space, $\Xi = \{\xi^h = (\xi_1^{h_1}, \ldots, \xi_m^{h_m})' \in \bigtimes_{i=1}^m \Xi_i : \|\xi^h\|_1 \leq N, h = (h_1, \ldots, h_m)', \|\xi\|_1 = \sum_{i=1}^m h_i\}$. The cardinality of Ξ is denoted by Ω. Moreover, we denote two generic states with $\xi^v = (\xi_1^{v_1}, \xi_2^{v_2}, \ldots, \xi_m^{v_m})'$ and $\xi^w = (\xi_1^{w_1}, \xi_2^{w_2}, \ldots, \xi_m^{w_m})'$. To simplify the complexity in the notation and with a slight abuse of notation, we indicate the state variable $\xi^h(j)$ at time j as $\xi(j)$, and we can use indistinctly $\xi_i^{h_i}$, ξ_i, and h_i.

• $U = \{c_1, \ldots, c_m, \nu\}$ is the overall action set. We denote with $U(\xi)$ the set of actions admissible at state ξ. We have

$$\begin{cases} U(\xi) = \{c_1, \ldots, c_m, \nu\}, & \text{if } \|\xi\|_1 < N \\ U(\xi) = \{\nu\}, & \text{if } \|\xi\|_1 = N. \end{cases} \tag{4}$$

We denote with $u(j)$ the control function at time j.

• $\mathcal{T}$ is the state transition mapping, represented by the state transition probability matrix with elements.

$$p_{\xi^v\xi^w}(u(j)) := p[\xi(j+1) = \xi^w | \xi(j) = \xi^v, u(j)]. \tag{5}$$

In case $u(j) = c_i$ we have

$$\begin{aligned} p_{\xi^v\xi^w}(c_i) &= p_{\xi_i^{v_i}\xi_i^{w_i}}(c_i) \times \prod_{\substack{l=1 \\ l \neq i}}^{m} p_{\xi_l^{v_l}\xi_l^{w_l}}(\nu) \\ &\text{s.t. } \max\{0, v_i - 1\} \leq w_i \leq \min\{N, v_i + 1\} \\ &\quad \max\{0, v_l - 1\} \leq w_l \leq v_l \end{aligned} \tag{6}$$

while if $u(j) = \nu$

$$\begin{aligned} p_{\xi^v\xi^w}(\nu) &= \prod_{l=1}^{m} p_{\xi_l^{v_l}\xi_l^{w_l}}(\nu) \\ &\text{s.t. } \max\{0, v_l - 1\} \leq w_l \leq \min\{N, v_l\}. \end{aligned} \tag{7}$$

We can also highlight the following aspects for the transition from the state ξ^v into the state ξ^w:

$$\max \|\xi^w\|_1 = \min(\|\xi^v\|_1 + 1, N) \tag{8}$$

$$\min \|\xi^w\|_1 = \max(0, \|\xi^v\|_1 - m). \tag{9}$$

• $R : \Xi \to [0, +\infty)$ is the reward function, defined as

$$R(\xi) = \sum_{i=1}^{m} c_i \xi_i. \tag{10}$$

Fig. 2 shows a resource allocation problem modelled as an MDP with number of resources $N = 2$, number of prices $m = 2$, and the action set $U = \{c_1, c_2, \nu\}$. Its generic state can be

denoted by $(\xi_1, \xi_2) \in \Xi = \{(0,0),(0,1),(1,0),(1,1),(0,2),(2,0)\}$, where ξ_1 and ξ_2 correspond to the number of resources allocated at price c_1 and c_2, respectively.

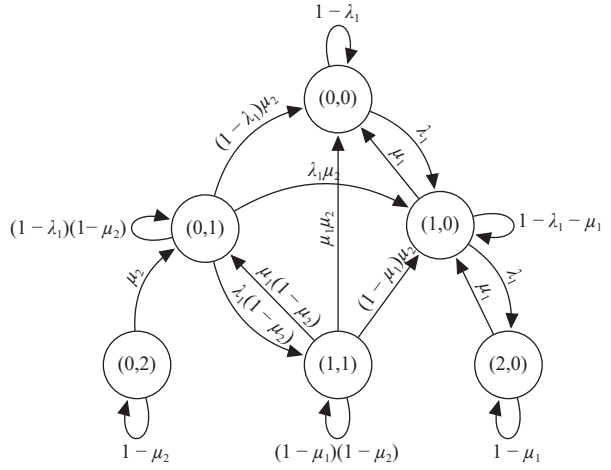

Fig. 2. An example of a resource allocation problem with $N = 2$ resources and prices c_1, c_2. The graph shows the entire state space and the state transition probabilities under the control $u = c_1$. As for the states $\{(2,0),(1,1),(0,2)\}$, the only admissible control is $u = \nu$.

C. Preliminaries on DP Techniques for Solving Resource Allocation Problems

Once a resource allocation problem is modelled as an MDP, we can apply DP techniques in order to solve the related decision making problem and calculate proper pricing policies.

To start with, let $\pi = \{u(0), \ldots, u(T-1)\}$ be the *policy*, that is to say, the sequence of control functions applied over the finite time horizon T. We define its value function $J_\pi : \Xi \to \mathbb{R}$ as follows:

$$J_\pi(\xi(0)) = E\left\{ J_T(\xi(T)) + \sum_{j=0}^{T-1} R(\xi(j+1)|\xi(j), u(j)) \right\} \tag{11}$$

where $E\{\cdot\}$ is the expectation operator, $J_T(\cdot)$ is the terminal value function, and $\xi(T)$ is the state at the terminal time T. The terminal value function is assumed to be known and bounded: in particular, in this paper, $J_T(\xi(T))$ is set to $R(\xi(T))$. Note that the expected operator $E\{\cdot\}$ is applied over the actual visited states, which determine the actual rewards collected over the finite time horizon T. This explains the operator "|" coming from the conditional probability notation used in the previous definition.

The value function of a specific policy $J_\pi(\cdot)$ can be regarded as the expected total revenue computed over the finite time horizon T for resource allocation problems, when applying a specific pricing policy. DP techniques aim at finding the optimal policy π^* that maximizes the expected total reward defined as

$$J^*(\xi(0)) = \max_\pi J_\pi(\xi(0)). \tag{12}$$

Under the assumption of having a relative small number of states, we can apply the exact DP algorithm, which exploits Bellman's principle of optimality [7]. In particular, starting from the terminal value function

$$J_T(\xi(T)) = R(\xi(T)) = \sum_{i=1}^{m} c_i \xi_i(T)$$

the optimal value function can be recursively expressed as

$$\begin{aligned} J_j^*(\xi(j)) &= \max_{u(j)\in\{c_1,c_2,\ldots,c_m,\nu\}} E\left\{ \sum_{i=1}^{m} c_i \xi_i(j) + J_{j+1}^*(\xi(j+1)) \right\} \\ &= \sum_{i=1}^{m} c_i \xi_i(j) + \max_{u(j)\in\{c_1,c_2,\ldots,c_m,\nu\}} E\left\{ J_{j+1}^*(\xi(j+1)) \right\} \\ &= \sum_{i=1}^{m} c_i \xi_i(j) + \max\Big[E\{J_{j+1}^*(\xi(j+1)) \,|u(j) = c_1\}, \\ &\qquad E\{J_{j+1}^*(\xi(j+1)) \,|u(j) = c_2\}, \ldots, \\ &\qquad E\{J_{j+1}^*(\xi(j+1)) \,|u(j) = c_m\}, \\ &\qquad E\{J_{j+1}^*(\xi(j+1)) \,|u(j) = \nu\} \Big]. \end{aligned}$$

The last expression comes from the expansion of the term $\max_{u(j)\in\{c_1,c_2,\ldots,c_m,\nu\}} E\{\cdot\}$. Thanks to Bellman's principle of optimality, the generated value function $J_j^*(\cdot)$ at each time slot j is equal to the optimal value function for the tail sub-problem from time j to time T. It is worth noting that the price chosen at the time slot j affects the state transitions from $\xi(j)$ to $\xi(j+1)$, and thus the expected value of the value function $J_{j+1}^*(\cdot)$. The value generated at the last step is equal to the optimal revenue $J^*(\xi(0))$ from the initial state $\xi(0)$.

As for real-world resource allocation problems, the applicability of exact DP methodologies is limited by severe scalability issues, due both to memory and computational requirements [20]. This phenomenon, known as curse of dimensionality, is caused by the cardinality of the system state space. In this regard, ADP approaches prove to offer powerful tools for addressing such scalability issues. The key idea to cope with the state space explosion is to substitute the original value function with a function having a more compact representation achieved by using a restricted set of selected features. Such a representation is also termed as approximation architecture [22], [23]. The linear architecture consists of approximating the value functions $J^*(\xi)$ and $J_\pi(\xi)$ as $\tilde{J}(\xi, r) = \phi(\xi)' r$, where $\phi(\xi) = [\phi_1(\xi), \phi_2(\xi), \ldots, \phi_q(\xi)]'$ is a vector of feature (or basis) functions evaluated over ξ and $r = [r_1, r_2, \ldots, r_q]'$ is a vector of parameters to be tuned by a suitable training process. In a more concise form and with a slight abuse of notation, for a value function $\tilde{J} \in \mathbb{R}^\Omega$ and a feature matrix $\Phi \in \mathbb{R}^{\Omega \times q}$ having $\phi'(\xi)$ as rows corresponding to each state $\xi \in \Xi$, we can set $\tilde{J} = \Phi r$.

III. Modeling Resource Allocation Problems With Specified Time Interval Reservation Requests

From a practical point of view, in resource allocation problems, there must be the possibility for customers to request resources in advance to satisfy their future needs. Due to the advance reservations, we have to consider that a portion of the resources can have already been allocated to customers

at specific time slots in the future. In this case, the number of available resources can change over time, thus the underlying allocation process has to be treated as a time-inhomogenous MDP (Ti-MDP).

More specifically, at the current time slot $k \geq 0$, a customer can ask for a resource by specifying a finite time interval $[h_1, h_2]$ for its utilisation (with $h_1 \geq k$). Such information determines both the allocation and release times of the required resources, the latter not subject to the customer stochastic behaviour. The minimum duration time $\tau = h_2 - h_1$ is considered as one time slot. A suitable price profile has to be proposed for maximising the expected total revenue. As shown later, such price profile can be composed of different prices, that is to say, the same resources can be allocated at different prices c_i over the given time interval. Booking in advance implies a significant upgrade of the MDP framework described in Section II: the number of the available resources N can change over time since some resources can be already assigned or released in advance, decreasing or increasing the set of resources subject to the system stochastic dynamics.

The formulation and the related resolution for resource allocation problems with instant and advance resource requests were firstly addressed by the authors in [17]. Hereafter, the followings requirements are additionally considered:

1) The proposed system manages the overbooking situations, which occur when the number of allocated resources at the (generic) time slot j can not be confirmed since new resources are already allocated at the time slot $j+1$ (due to the advance resource reservation mechanism), and the overall needs can not be satisfied by the system overall capacity of N resources. This overbooking situation occurs since the system handles both instant and future resource requests.

2) Customers can hold the resources to the necessary extent only if the system capacity allows it: already-reserved resources for the next time slot can cause an overbooking situation.

3) In such circumstances, the price manager forces the customers with the lowest growth rates (as defined later in the paper) to release their own resources (or some of them) in order to fulfill the reservations of the next time slot under the system capacity constraint.

As shown in the previous section, resource allocation problems with no advance resource requests can be modelled by means of a time-homogeneous MDP, featured by a stationary state transition mapping $\mathcal{T}$. In this section, the system C^j is introduced, with the superscript j indicating that its underlying MDP dynamics become time-variant due to the already reserved resources (note that we denote with j a generic time slot over the finite time horizon T).

As shown in this section, state transitions between two consecutive time slots j and $j+1$ may occur either stochastically or deterministically, the latter depending on the set of reservations for such two consecutive time slots. Unlike [17], the set of actions, denoted by $\bar{U}$, has the additional action ψ, i.e., $\bar{U} = \{c_1, \ldots, c_m, \nu, \psi\}$. Such control ψ is applied whenever the price manager has to force customers to release their own resources in order to solve an overbooking situation. Moreover, we provide a formulation for resource allocation problems and their corresponding resolution algorithms suitable for computer based implementation.

Prior to addressing the modeling, it is necessary to define some terms and notation.

Definition 1: A *reservation request* occurs whenever the customer at the current time slot k asks for allocation of a resource in advance for the future time slot $h > k$.

Definition 2: We call the price c_i a *reservation price* if the decision maker allocates a resource with the price c_i for the time slot $h > k$ in response of *reservation request* at the current time slot k.

Definition 3: We define the term $(\lambda_i - \mu_i)$ as the growth rate associated with the price c_i, where λ_i and μ_i are allocation and deallocation probability requests, respectively.

Definition 4: We define the *set of reservations* for different time slots as

$$x = \{\ldots, x^{j-1}, x^j, x^{j+1}, \ldots\} \tag{13}$$

where $x^j \in \mathbb{N}_0^m, j \in \mathbb{N}$, the i-th element of x^j, namely x_i^j, is the number of reserved resources at the price c_i at the generic time slot j. The vector x^j is referred to as *reservation vector*.

The Ti-MDP for resource allocation problems with advance reservations can be defined by the tuple $C^j = \langle \Xi^j, x^j, \bar{U}, \mathcal{T}^j, R \rangle$, where

- Ξ^j is the state space at the generic time slot j, $\Xi^j = \{\bar{\xi}^\iota = (\bar{\xi}_1^{\iota_1}, \ldots, \bar{\xi}_m^{\iota_m})' \in \bigtimes_{i=1}^m \Xi_i : \iota_i \geq x_i^j \ \& \ \|\bar{\xi}^\iota\|_1 \leq N\}$. To simplify the complexity in the notation with a slight abuse of notation, we denote $\bar{\xi} := \bar{\xi}^\iota$ as a generic state of the process. We can use indistinctly $\bar{\xi}_i^{\iota_i}$, $\bar{\xi}_i$, and ι_i. Moreover, we denote with $\bar{\xi}(j) \in \Xi^j$ the state variable at time slot j, $|\Xi^j| = \Omega^j$, and $\Xi^j \subseteq \Xi$.
- $x^j \in \mathbb{N}_0^m$ is the *reservation vector* having the constraint

$$0 \leq \|x^j\|_1 \leq N. \tag{14}$$

- $\bar{U} = \{c_1, \ldots, c_m, \nu, \psi\}$ is the overall input set. Two general cases can happen:

◇ $\|x^j\|_1 \leq N, \|x^{j+1}\|_1 = N$

$$\bar{U}(\bar{\xi}) = \{\nu, \psi\} \tag{15}$$

◇ $\|x^j\|_1 \leq N, \|x^{j+1}\|_1 < N$

$$\bar{U}(\bar{\xi}) = \{c_1, \ldots, c_m, \nu, \psi\}. \tag{16}$$

We denote with $\bar{u}(j)$ the control function at time j.

- $\mathcal{T}^j$ is the state transition mapping, computed by applying the following composition (note that $\rightarrow$ means function, while $\mapsto$ mapping)

$$\mathcal{T}^j : \Xi^j \xrightarrow{\mathcal{F}} \Xi \overset{\bar{U}}{\mapsto} \Xi \xrightarrow{\mathcal{G}} \Xi^{j+1} \tag{17}$$

where the functions $\mathcal{F}$ and $\mathcal{G}$ are

$$\mathcal{F} : \Xi^j \rightarrow \Xi$$
$$\mathcal{F}(\bar{\xi}(j)) = \{\xi(j) \in \Xi : \xi_i(j) = \bar{\xi}_i(j) - x_i^j\} \tag{18}$$

$$\mathcal{G} : \Xi \rightarrow \Xi^{j+1}$$
$$\mathcal{G}(\xi(j)) = \{\bar{\xi}(j+1) \in \Xi^{j+1} : \bar{\xi}_i(j+1) = \xi_i(j) + x_i^{j+1}\}. \tag{19}$$

At each time slot j, we remove from the state variable $\bar{\xi}(j)$ the reservation vector x^j by applying the subtraction function $\mathcal{F}$, and we add the reservation vector x^{j+1} of the next time slot $j+1$ by applying the addition function $\mathcal{G}$.

The elements of the state transition mapping $\mathcal{T}^j$ are denoted by $p_{\bar{\xi}^v(j)\bar{\xi}^w(j+1)}(x^j,\bar{u}(j))$, with $\bar{\xi}^v(j)\in\Xi^j$ and $\bar{\xi}^w(j+1)\in\Xi^{j+1}$. For the sake of simplicity, the dependency on the time slot and the reservation vector can be removed, thus we have, respectively: $p_{\bar{\xi}^v\bar{\xi}^w}(x^j,\bar{u})$ and $p_{\bar{\xi}^v\bar{\xi}^w}(\bar{u})$.

It is assumed that the admissible control $\bar{u}(j)\in\bar{U}$ is chosen such that the price manager gives the priority to the reservations x^{j+1}. By denoting with $O_m\in\mathbb{R}^m$ the vector having all 0 as components, the dynamics of the state variable $\bar{\xi}(j)$ can be described as follows (note that the superscripts v and w are used to indicated the state variable values at j and $j+1$, respectively):

◇ For $\|x^j\|_1\le N$, $\|x^{j+1}\|_1=N$ (similarly for the action ψ),

$$\begin{aligned}&\mathcal{T}^j:\Xi^j\xrightarrow{\mathcal{F}}O_m\xrightarrow{\nu}O_m\xrightarrow{\mathcal{G}}\Xi^{j+1},\text{and}\\&p_{\bar{\xi}^v\bar{\xi}^w}(\nu)=1.\end{aligned}\tag{20}$$

Note that the state transition from $\bar{\xi}(j)=\bar{\xi}^v$ into $\bar{\xi}(j+1)=\bar{\xi}^w$ is deterministic, and that $\bar{\xi}^w=x^{j+1}$.

◇ For $\|x^j\|_1<N$, $\|x^{j+1}\|_1<N$,

$$\mathcal{T}^j:\Xi^j\xrightarrow{\mathcal{F}}\Xi\mapsto\Xi\xrightarrow{\mathcal{G}}\Xi^{j+1},\text{and}\tag{21}$$

1) if $\|\bar{\xi}(j)\|_1<N$,

$$p_{\bar{\xi}^v\bar{\xi}^w}(\bar{u}(j)):=p[\xi(j+1)=\xi^w|\xi(j)=\xi^v,\bar{u}(j)]\tag{22}$$

such that, if $\bar{u}(j)=c_i$, or $\bar{u}(j)=\nu$, then

$$p_{\bar{\xi}^v\bar{\xi}^w}(c_i)=p_{\xi_i^{v_i}\xi_i^{w_i}}(c_i)\times\prod_{\substack{l=1\\l\neq i}}^{m}p_{\xi_l^{v_l}\xi_l^{w_l}}(\nu),\text{or}\tag{23}$$

$$p_{\bar{\xi}^v\bar{\xi}^w}(\nu)=\prod_{l=1}^{m}p_{\xi_l^{v_l}\xi_l^{w_l}}(\nu),\text{respectively.}\tag{24}$$

2) If none of the controls $\{c_1,\ldots,c_m,\nu\}$ are admissible at the current state $\bar{\xi}(j)$, then the price manager forces some customers with lower growth rates to release their resources. In this case, we have

$$\mathcal{T}^j:\Xi^j\xrightarrow{\mathcal{F}}\Xi\xmapsto{\psi}\Xi\xrightarrow{\mathcal{G}}\Xi^{j+1}.\tag{25}$$

The transition (25) is performed deterministically since the customers are forced to release the resources.

◇ For $\|x^j\|_1=N$, $\|x^{j+1}\|_1<N$, if $\bar{u}(j)=c_i$, or $\bar{u}(j)=\nu$,

$$\begin{aligned}&\mathcal{T}^j:\Xi^j\xrightarrow{\mathcal{F}}O_m\xrightarrow{\bar{U}}O_m\xrightarrow{\mathcal{G}}\Xi^{j+1}\\&p_{\bar{\xi}^v\bar{\xi}^w}(c_i)=p_{\xi_i^{v_i}\xi_i^{w_i}}(c_i)\times\prod_{\substack{l=1\\l\neq i}}^{m}p_{\xi_l^{v_l}\xi_l^{w_l}}(\nu)\\&p_{\bar{\xi}^v\bar{\xi}^w}(\nu)=\prod_{l=1}^{m}p_{\xi_l^{v_l}\xi_l^{w_l}}(\nu).\end{aligned}\tag{26}$$

◇ For $\|x^j\|_1=N$, $\|x^{j+1}\|_1=N$, the admissible control is ν for any resource request for the time slot $j+1$. The transition is therefore

$$\begin{aligned}&\mathcal{T}^j:\Xi^j\xrightarrow{\mathcal{F}}O_m\xrightarrow{\nu}O_m\xrightarrow{\mathcal{G}}\Xi^{j+1}\\&p_{\bar{\xi}^v\bar{\xi}^w}(\nu)=1.\end{aligned}\tag{27}$$

• $R:\Xi^j\to[0,+\infty)$ is the reward function, defined in this case as

$$R(\bar{\xi}(j))=\sum_{i=1}^{m}c_i\bar{\xi}_i(j).\tag{28}$$

Example 1: Consider a resource allocation problem with $m=2$, $N=2$ and the following two cases:

• $x^j=(1,1)'$ and $x^{j+1}=(0,1)'$;

• $x^j=(0,0)'$ and $x^{j+1}=(0,1)'$.

1) If $x^j=(1,1)'$ and $x^{j+1}=(0,1)'$, based on the Ti-MDP definition, the only feasible state at the time slot j is $\bar{\xi}(j)=(1,1)$. By applying $\mathcal{F}$, we have $\bar{\xi}(j)=(1,1)\xrightarrow{\mathcal{F}}\xi(j)=(0,0)$. By using the formulas in (26) with admissible controls c_i and ν, we have

• If $\bar{u}(j)=c_1$,

$$\xi(j)=(0,0)\overset{c_1}{\mapsto}\begin{bmatrix}(0,0)\\(1,0)\end{bmatrix}\overset{probabilities}{\Longrightarrow}\begin{bmatrix}(1-\lambda_1)\\(\lambda_1)\end{bmatrix}$$

and by applying the function $\mathcal{G}$

$$\begin{bmatrix}(0,0)\\(1,0)\end{bmatrix}\xrightarrow{\mathcal{G}}\bar{\xi}(j+1)=\begin{bmatrix}(0,1)\\(1,1)\end{bmatrix}.$$

• If $\bar{u}(j)=c_2$,

$$\xi(j)=(0,0)\overset{c_2}{\mapsto}\begin{bmatrix}(0,0)\\(0,1)\end{bmatrix}\overset{probabilities}{\Longrightarrow}\begin{bmatrix}(1-\lambda_2)\\(\lambda_2)\end{bmatrix}$$

and by applying the function $\mathcal{G}$

$$\begin{bmatrix}(0,0)\\(0,1)\end{bmatrix}\xrightarrow{\mathcal{G}}\bar{\xi}(j+1)=\begin{bmatrix}(0,1)\\(0,2)\end{bmatrix}.$$

• If $\bar{u}(j)=\nu$,

$$\xi(j)=(0,0)\overset{\nu}{\mapsto}[(0,0)]\overset{probability}{\Longrightarrow}[1]$$

and by applying the function $\mathcal{G}$

$$[(0,0)]\xrightarrow{\mathcal{G}}\bar{\xi}(j+1)=[(0,1)].$$

It can be noted that all the controls are admissible for the state $\bar{\xi}(j)=(1,1)$ thanks to the reservation vectors for the two subsequent time slots.

2) If $x^j=(0,0)'$ and $x^{j+1}=(0,1)'$, based on the Ti-MDP definition, all the states at the time slot j are feasible since the reservation vector is $x^j=(0,0)$. However, there exist some states (e.g., $(0,2)$), where none of the controls $\{c_1,\ldots,c_m,\nu\}$ are admissible and the price manager has to apply the control ψ.◇.

IV. DP Framework for Solving Resource Allocation Problems With Advance Resource Requests

This section introduces the DP framework used to compute a proper pricing policy for resource allocation problems with advance resource requests. Such framework along with its related definitions is used in the next section, where the reservation pricing algorithm is described.

We denote with $J_j^{*,x^j}(\bar{\xi}(j))$ the optimal value function at the

generic time slot j, starting from any state $\bar{\xi}(j) \in \Xi^j$ and with the reservation vector x^j. By exploiting Bellman's principle of optimality, it is possible to write the following expression:

$$J_j^{*,x^j}(\bar{\xi}(j)) = \max_{\bar{u}(j)\in\{c_1,\ldots,c_m,\nu,\psi\}} E\left[\sum_{i=1}^{m} c_i\bar{\xi}_i(j) + J_{j+1}^{*,x^{j+1}}(\bar{\xi}(j+1))\right]. \tag{29}$$

The control function which satisfies (29) is denoted by $\bar{u}^*(j)$. The previous expression is the core of the exact DP algorithm [7], and allows to calculate the optimal policy $\bar{\pi}^* = \{\bar{u}^*(0),\ldots,\bar{u}^*(T-1)\}$ over the whole time horizon T (note that we assume that the underlying Ti-MDP is computationally tractable). With the knowledge of the reservation vectors for the entire planning horizon time, the price manager can apply the calculated optimal policy $\bar{\pi}^*$. Since the exact DP algorithm is applied, such optimal policy exists and is unique. In the following, such solution is referred to as DP off-line solution. At each time slot j, in case of changes in the future reservation vectors, it is required to recompute the DP off-line solution. This recursive refresh of the exact DP algorithm results links the concepts of the MPC to the DP for resource allocation problems [13].

Prior to illustrating the reservation pricing algorithm, we need to discuss some further aspects and introduce the related notation. The *transition probability matrix* $P^*(x^j, j) \in \mathbb{R}_+^{\Omega\times\Omega}$ at a generic time slot $j \geq k$ for the C^j system (when controlled by the policy $\bar{\pi}^*$) is defined as follows:

$$P^*(x^j, j) := \begin{cases} [p_{\bar{\xi}^v\bar{\xi}^w}(x^j, \bar{u}^*(j))],\ \bar{\xi}_i^v(j) \geq x_i^j,\ \bar{\xi}_i^w(j+1) \geq x_i^{j+1} \\ [0], \qquad\qquad \text{otherwise} \end{cases} \tag{30}$$

where $p_{\bar{\xi}^v\bar{\xi}^w}(x^j, \bar{u}^*(j)) = p_{\bar{\xi}^v\bar{\xi}^w}(\bar{u}^*(j))$ and $\mathbb{R}_+$ stands for the set of non-negative real numbers. The transition probabilities associated with the infeasible states $\bar{\xi}_i(j) < x_i^j$ are set to 0. With a slight abuse of notation, it is noticed that the right hand side of (30) includes all the elements of the *transition probability matrix*, and that the dependency on the time slot in $p_{\bar{\xi}^v(j)\bar{\xi}^w(j+1)}(\cdot)$ is removed.

The *probability distribution vector* $\varepsilon^*(x^j, j) \in \mathbb{R}_+^{\Omega}$ for the C^j system is defined as follows:

$$\varepsilon^*(x^j, j) := \begin{cases} [\varepsilon^*_{\bar{\xi}^v}(x^j, j)],\ \bar{\xi}_i^v(j) \geq x_i^j \\ [0], \qquad\qquad \text{otherwise} \end{cases} \tag{31}$$

where $\varepsilon^*_{\bar{\xi}^v}(x^j, j)$ stands for the probability of being at the state $\bar{\xi}^v$ at the time slot j (with $\|\varepsilon^*(x^j, j)\|_1 = 1$). It is assumed that the C^j system is controlled by the policy $\bar{\pi}^*$ and that the initial condition $\bar{\xi}(k)$ is set to $\bar{\xi}^w \in \Xi^k$. In other words, we have

$$\begin{cases} \varepsilon_{\bar{\xi}^v}(x^k, k) = 1 & \bar{\xi}(k) = \bar{\xi}^w \\ \varepsilon_{\bar{\xi}^w}(x^k, k) = 0 & \bar{\xi}(k) \neq \bar{\xi}^w. \end{cases}$$

The probability of being at the infeasible states $\bar{\xi}_i(j) < x_i^j$ is set to 0. It is noticed that the right hand side of (31) includes all the elements of the probability distribution vector.

The following relation holds between the probability distribution vector and the transition probability matrix $P^*(x^j, j)$:

$$\varepsilon^*(x^{h-1}, h-1)' = \varepsilon(x^k, k)' \times \prod_{j=k+1}^{h-2} P^*(x^j, j). \tag{32}$$

It follows from the property of Markov chains, i.e., for a given initial probability distribution of the process one can calculate the probability distribution of the MDP for an arbitrary time slot in the future [21]. In case of large state spaces, one can execute the policy $\bar{\pi}^*$ and use Monte Carlo simulations to approximate the values of the probability distribution vector [7], [20].

We define $e_{c_i} \in \mathbb{N}_0^m$ as the unitary vector having 1 in the i-th entry (corresponding to the price c_i) and 0 in all the others. Additionally, we denote by $e_{c_i}^j$ the unitary vector e_{c_i} at the time slot j. We introduce the auxiliary vector $\mathcal{V}^j : U \to \mathbb{N}_0^m$ at the time slot j as

$$\mathcal{V}^j(u) = \begin{cases} x^j + e_{c_i}^j, & u \in \{c_1,\ldots,c_m\} \\ x^j, & u = \nu \end{cases} \tag{33}$$

and indicate by $\mathcal{V}_i^j$ an element of this vector.

A. Applying the Principle of Optimality for the Advance Resource Reservations

Let us suppose that the price manager has to allocate a new resource for its future utilisation at the time slot $h > k$ and that the DP offline solution $\bar{\pi}^*$ is available. By focusing on the time interval $[k, h-1]$ and by exploiting Bellman's principle of optimality [7], the optimal value function $J_k^{*,x^k}(\bar{\xi}(k))$ related to such DP offline solution (computed at the current state $\bar{\xi}(k)$) can be expressed and developed as follows:

$$\begin{aligned} J_k^{*,x^k}(\bar{\xi}(k)) &= \max_{\bar{u}^*(j)\in\{c_1,\ldots,c_m,\nu,\psi\}} E\Bigg\{R(\bar{\xi}(k)) \\ &+ \sum_{j=k}^{h-3} R(\bar{\xi}(j+1)|\bar{\xi}(j), \bar{u}(j)) + J_{h-1}^{*,x^{h-1}}(\bar{\xi}(h-1))\Bigg\} = R(\bar{\xi}(k)) \\ &+ \max_{\bar{u}(j)\in\{c_1,\ldots,c_m,\nu,\psi\}} E\Bigg\{\sum_{j=k}^{h-3} R(\bar{\xi}(j+1)|\bar{\xi}(j), \bar{u}(j)) + J_{h-1}^{*,x^{h-1}}(\bar{\xi}(h-1))\Bigg\} \\ &= R(\bar{\xi}(k)) + E\Bigg\{\sum_{j=k}^{h-3} R(\bar{\xi}(j+1)|\bar{\xi}(j), \bar{u}^*(j))\Bigg\} \\ &+ \max_{\bar{u}(h-1)\in\{c_1,\ldots,c_m,\nu\}} E\Big\{R(\bar{\xi}(h-1)|\bar{\xi}(h-2), \bar{u}^*(h-2)) + J_h^{*,x^h}(\bar{\xi}(h))\Big\}. \end{aligned}$$

This means that the DP offline solution $\bar{\pi}^*$ can be applied from the current time slot k up to the time slot $h-2$: in other words, the decisions made during the time interval $[k, h-2]$ can be decoupled from the one made at the time slot $h-1$. As a consequence, the reservation pricing algorithm evaluates how each possible price for the additional resource can affect the value function $J_k^{*,x^k}(\bar{\xi}(k))$. Then, the algorithm selects the action maximizing the computed value function.

From an implementation perspective, this implies applying the DP offline solution $\bar{\pi}^*$ up to $h-2$ (by using Monte Carlo simulations in case of large state spaces) and compute the new value function at $\bar{\xi}(k)$ for the different possible decisions. In this regard, we define $J_k^{\mathcal{V}^h(u)}(\bar{\xi}(k))$ as the value function calculated at the initial state $\bar{\xi}(k)$, when the policy $\bar{\pi}_k^* = \{\bar{u}^*(k),\ldots,\bar{u}^*(h-2)\}$ is applied and a decision $u \in U = \{c_1,\ldots,c_m\}$ at the time slot $h-1$ is selected to allocate

the new resource at the time slot h. More specifically, $J_k^{\mathcal{V}^h(u)}(\bar{\xi}(k))$ can be defined and expanded as follows:

$$\begin{aligned} J_k^{\mathcal{V}^h(u)}(\bar{\xi}(k)) &= R(\bar{\xi}(k)) \\ &+ E\left\{\sum_{j=k}^{h-3} R(\bar{\xi}(j+1)|\bar{\xi}(j),\bar{u}^*(j))\right\} \\ &+ E\left\{\bar{R}(\bar{\xi}(h-1)|\bar{\xi}(h-2),\bar{u}^*(h-2)) + J_h^{\mathcal{V}^h(u)}(\bar{\xi}(h))\right\} \\ &= R(\bar{\xi}(k)) + E\left\{\sum_{j=k}^{h-3} R(\bar{\xi}(j+1)|\bar{\xi}(j),\bar{u}^*(j))\right\} \\ &+ \Bigg[\sum_{\bar{\xi}(h-1)\in\Xi^{h-1}} \varepsilon_{\bar{\xi}}^*(\mathcal{V}^{h-1}(v),h-1) \\ &\times \Big\{R(\bar{\xi}(h-1)|\bar{\xi}(h-2),\bar{u}^*(h-2)) \\ &+ \sum_{\bar{\xi}(h)\in\Xi^h} p_{\bar{\xi}(h-1)\bar{\xi}(h)}(\mathcal{V}^{h-1}(v),u) J_h^{\mathcal{V}^h(u)}(\bar{\xi}(h))\Big\}\Bigg] \end{aligned}$$

$$\text{s.t. } \|\bar{\xi}(j)\|_1 \geq \|\mathcal{V}^j(v)\|_1,\ j\in[k,h-1]$$
$$\|\bar{\xi}(h)\|_1 \geq \|\mathcal{V}^h(u)\|_1. \tag{34}$$

It is worth highlighting that:

• Compared with the previous expression, $J_h^{*,x^h}(\bar{\xi}(h))$ has been replaced with $J_h^{\mathcal{V}^h(u)}(\bar{\xi}(h))$. The *max* operator has been also removed since a specific action $u\in U$ is chosen at the time slot $h-1$.

• The probability distribution vector $\varepsilon_{\bar{\xi}}^*(\mathcal{V}^{h-1}(v),h-1))$ can be calculated by using the (31). Note that $\mathcal{V}^j(v)=x^j$ when $j\in[k,h-1]$.

• The function $J_h^{\mathcal{V}^h(u)}(\bar{\xi}(h))$ can be computed as follows:

$$\begin{aligned} J_h^{\mathcal{V}^h(u)}(\bar{\xi}(h)) &= R(\bar{\xi}(h)) \\ &+ \sum_{\forall\bar{\xi}(h+1)\in\Xi^{h+1}} p_{\bar{\xi}(h)\bar{\xi}(h+1)}(\mathcal{V}^h(u),\bar{u}^*(h)) J_{h+1}^*(\bar{\xi}(h+1)) \end{aligned}$$

$$\text{s.t. } \|\bar{\xi}(h)\|_1 \geq \|\mathcal{V}^h(u)\|_1. \tag{35}$$

It can be noted that: i) the action u for allocating the new resource affects the state $\bar{\xi}(h)$ (see also (34)); ii) the DP off-line solution is applied at the time slot h; iii) the optimal value function $J_{h+1}^*(\cdot)$ is used.

V. Reservation Pricing Algorithm for Specified Time Interval Reservation Requests

This section describes the algorithm for calculating the price profile to be proposed by the price manager in case of a reservation request for the future time interval $[h_1,h_2]$.

The following assumptions are made:

• The current system state $\bar{\xi}(k)$ is known to the decision maker.

• The initial reservation vectors x^j for all the time slots $j\in[0,T-1]$ are provided. Note that, as time goes by, these vectors can be updated since customers can request resources in advance to satisfy their future needs.

• The DP off-line solution $\bar{\pi}^*$ is available over the finite time horizon $[0,T-1]$. In this regard, we remind that $k,h_1,h_2\in[0,T-1]$. Like the reservation vectors, the policy $\bar{\pi}^*$ can be recomputed, as shown in the algorithm.

• Without any loss of generality, the underlying Ti-MDP is computationally tractable, that is to say, ADP and Monte Carlo simulations are not needed.

In the algorithm, a backwards iteration strategy is proposed. More specifically, the whole time interval is scanned backwards, and for each $h\in[h_1,h_2]$ the best price is calculated. At the same time, the temporary set of reservations is updated along with the associated optimal value function and policy. Finally, the resulting price profile is proposed to the customer. In case the customer accepts it, the new set of reservations is confirmed, as well as the associated optimal value function and policy.

The reservation pricing algorithm consists of the following steps (at the beginning $k=0$):

1) Apply the exact DP algorithm over the finite horizon $[k,T-1]$ for the Ti-MDP with the tuple $C^j=\langle\Xi^j,x^j,\bar{U},\mathcal{T}^j,R\rangle$ and $j\in[k,T-1]$, before processing the new resource request at the current time k.

2) Save $\bar{u}^*(j)$ and the optimal value functions $J_j^{*,x^j}(\bar{\xi}(j))$ as the tabular representation for the time interval $j\in[k,T-1]$ based on the previous step.

3) Set $h=h_2$

4) Set the auxiliary variable $g^j=x^j$ for the interval $j\in[k,h_2]$.

5) While $h\geq h_1$

◇ Set $\mathcal{V}^j(v)=g^j$ for the time interval $j\in[k,h-1]$.

◇ For each $u\in U$ calculate $J_k^{\mathcal{V}^h(u)}(\bar{\xi}(k))$ as described in Section IV-A.

◇ Compute the optimal reservation price for the time slot h as follows:

$$c_i^* = \arg\max_{u\in\bar{U}} J_k^{\mathcal{V}^h(u)}(\bar{\xi}(k)).$$

◇ $g^h \leftarrow g^h + e^h_{c_i^*}$.

◇ Update the price profile $\underline{c}=\{c_{h_1},\ldots,c_{h_2}\}$ with $c_h=c_i^*$.

◇ Apply the exact DP algorithm for the Ti-MDP with the tuple $C^j=\langle\Xi^j,g^j,\bar{U},\mathcal{T}^j,R\rangle$ over the time interval $j\in[k,h]$, and save the results.

◇ Set $h\leftarrow h-1$.

6) The optimal value function $J_k^{*,g^k}(\bar{\xi}(k))$, which is calculated at the last iteration $h=h_1$, takes into account the price profile over the time intervals $h\in[h_1,h_2]$.

7) Calculate the following expected value:

$$J_{E,k}^{\tau}(\bar{\xi}(k)) = \gamma_{\text{acc}}\, J_k^{*,g^k}(\bar{\xi}(k)) + (1-\gamma_{\text{acc}})\, J_k^{*,x^k}(\bar{\xi}(k)) \tag{36}$$

where γ_{acc} is the acceptance probability of the proposed price profile from the customer side over the entire time interval $[h_1,h_2]$.

8) The price manager accepts or rejects the reservation request by comparing these two values:

$$\max_J \left\{J_{E,k}^{\tau}(\bar{\xi}(k)), J_k^{*,x^k}(\bar{\xi}(k))\right\} \tag{37}$$

where $J_k^{*,x^k}(\bar{\xi}(k))$ is the expected optimal value function in case the reservation request is rejected.

9) If fulfilling the reservation request is more profitable than rejecting it (using (37)), the price manager offers the price profile $\underline{c}$.

10) If the customer accepts the proposed price profile, set $x^j \leftarrow g^j$ for the entire interval $j \in [h_1, h_2]$.

11) Set $k \leftarrow k+1$ and go to Step 1.

It is easy to verify that single time slot reservation requests can be derived from the above algorithm. The stochastic prediction of the system evolution (up to the time where the new resource is requested), the effect of the possible prices on the current expected total revenue, and the renewal of the policy pricing (in case of the customer's acceptance) shows a strong link between MPC and DP, as already discussed in [13].

VI. Modeling Resource Allocation Problems With Specified and Unspecified Time Interval Reservation Requests

In the normal course of events, it may happen that customers request resources for future utilization without specifying their release time. As a result, unlike the allocation time $h > k$, such resources are released according to the system stochastic dynamics, that is to say, the release event is linked to the death rate μ_i of the proposed price. This section outlines the modeling of the underlying Ti-MDP with specified and unspecified time interval reservation vectors, as well as the associated pricing algorithm.

Let s be the set of unspecified time interval reservations. In particular, we define

$$s = \{\dots, s^{j-1}, s^j, s^{j+1}, \dots\} \tag{38}$$

where $s^j \in \mathbb{N}_0^m$, $j \in \mathbb{N}$, and the i-th element of s^j, namely s_i^j, is the number of booked resources at the price c_i at the generic time instant j.

The dynamic of the underlying Ti-MDP for resource allocation problems with specified and unspecified time interval reservation requests can be defined by the tuple $C^j = \langle \Xi^j, x^j, s^j, \bar{U}, \mathcal{T}^j, R \rangle$, where

- Ξ^j is the state space at the time slot j, $\Xi^j = \{\bar{\xi}^\iota = (\xi_1^{\iota_1}, \dots, \xi_m^{\iota_m}) \in \bigtimes_{i=1}^m \Xi_i : \iota_i \geq x_i^j + s_i^j \ \& \ \|\bar{\xi}^\iota\|_1 \leq N\}$, where $\xi^\iota = (\xi_1^{\iota_1}, \xi_2^{\iota_2}, \dots, \xi_m^{\iota_m})$. To simplify the complexity in the notation, we denote $\bar{\xi} := \bar{\xi}^\iota$ as a generic state of the process; the same for $\bar{\xi}_i := \bar{\xi}_i^{\iota_i}$. Moreover, we denote with $\bar{\xi}(j)$ the state variable at time slot j, $|\Xi^j| = \Omega^j$, $\Xi^j \subseteq \Xi$.
- $x^j, s^j \in \mathbb{N}_0^m$ are the reservation vectors having the constraint

$$0 \leq \|x^j + s^j\|_1 \leq N. \tag{39}$$

Note that these vectors are considered deterministic.

- $\bar{U} = \{c_1, \dots, c_m, \nu, \psi\}$ is the overall input set. Two general cases can happen

◇ $\|x^j + s^j\|_1 \leq N$, $\|x^{j+1} + s^{j+1}\|_1 = N$

$$\bar{U}(\bar{\xi}) = \{\nu, \psi\} \tag{40}$$

◇ $\|x^j + s^j\|_1 \leq N$, $\|x^{j+1} + s^{j+1}\|_1 < N$

$$\bar{U}(\bar{\xi}) = \{c_1, \dots, c_m, \nu, \psi\}. \tag{41}$$

We denote with $\bar{u}(j)$ the input at time j.

- $\mathcal{T}^j$ is the state transition mapping, which is defined as follows:

$$\mathcal{T}^j : \Xi^j \xrightarrow{\mathcal{F}} \Xi \mapsto \Xi \xrightarrow{\bar{U}} \Xi \xrightarrow{\mathcal{B}} \Xi^{j+1} \tag{42}$$

where the function $\mathcal{B}$ is

$$\mathcal{B} : \Xi \to \Xi^{j+1}$$
$$\mathcal{B}(\xi(j)) = \{\bar{\xi}(j+1) \in \Xi^{j+1} : \bar{\xi}_i(j+1) = \xi_i(j) + x_i^{j+1} + s_i^{j+1}\}. \tag{43}$$

The state transition probabilities can be derived in the same way of specified time interval reservation requests. The only differences are the replacement of the function $\mathcal{G}$ with $\mathcal{B}$ and the new condition on the reservation vectors, i.e., $0 \leq \|x^j + s^j\|_1 \leq N$.

- $R : \Xi^k \to [0, +\infty)$ is the reward function, defined as

$$R(\bar{\xi}(j)) = \sum_{i=1}^m c_i \bar{\xi}_i(j). \tag{44}$$

A. Reservation Pricing Algorithm for Solving Resource Allocation Problems With Specified and Unspecified Time Interval Requests

The reservation pricing algorithm shown in Section V can be easily extended to handle resource requests with unspecified time intervals. In particular, at the current time slot k, three cases can be managed, that is to say, the one for the instant reservation requests (where the proposed price is given by the DP offline solution $\bar{\pi}^*$), the one for the advance reservations with specified time interval, and the one for the advance reservations with unspecified time interval.

As for the latter, the main difference derives from the fact that only the allocation time h for the new resource is provided. Thus, the algorithm has to evaluate how each possible price used to allocate such resource at the only time slot h affects the current optimal value function. Besides that, some notation has to be adapted, e.g.,

- The optimal value function defined in (29) is denoted by $J_j^{*,x^j s^j}(\bar{\xi}(j))$.
- The auxiliary vector $\mathcal{V}^j : U \to \mathbb{N}_0^m$ at the time slot j becomes

$$\mathcal{V}^j(u) = \begin{cases} x^j + s^j + e_{c_i}^j, & u \in \{c_1, \dots, c_m\} \\ x^j + s^j, & u = \nu. \end{cases} \tag{45}$$

VII. Reservation Pricing Approach for Resource Allocation Problems With Large State Space

This section addresses resource allocation problems with large state space. In particular, we assume that the underlying Ti-MDP is not computationally tractable, thus ADP and Monte Carlo simulations are needed. More specifically, this section outlines the ADP based approach used for solving resource allocation problems with a large state space, and shows the changes to the reservation pricing algorithm described in Section V.

The USLS-ADP algorithm is adopted to compute an approximation of the optimal value function $J_j^{*,x^j,s^j}(\cdot)$ over the

finite time horizon $[0, T-1]$. Such algorithm belongs to the value function approximation category [8] and was presented by the authors in [19], in a different context. The reader can refer to such paper for more details about its theoretical aspects (e.g., its contraction mapping and the monotonicity properties) and its convergence results over an infinite time horizon. It is worth highlighting that the USLS-ADP algorithm is used over a finite time horizon in this paper.

Hereafter, we apply the USLS-ADP algorithm to solve the curse of dimensionality for resource allocation problems with instant and future resource requests. For the sake of simplicity, we drop the notation concerning the future reservation requests, e.g., the reservation vectors x^j and s^j.

Consider a low-dimensional approximation of $J_j^* \in \mathbb{R}^{\Omega}$ that has the form (note that the explicit dependency on the state has been removed as well)

$$J_j^* \approx \Phi r_j^*, \quad j = 0, \ldots, T-1 \tag{46}$$

where $\Phi \in \mathbb{R}^{\Omega \times q}$ is the feature matrix whose columns are the basis functions, and $r_j^* \in \mathbb{R}^q$ is the parameter vector that has to be computed ($q << \Omega$). We define by $\tilde{J}_j^* = \Phi r_j^*$ the approximate value of vector J_j^*. This leads us to the following weighted Euclidean least squares minimization problem at each time slot j:

$$r_j^* = \arg\min_{r_j \in \mathbb{R}^q} \|J_j^* - \Phi r\|_{\epsilon}^2 \tag{47}$$

where $\epsilon \in \mathbb{R}_+^{\Omega}$ and $\|\epsilon\|_1 = 1$. These weights show the importance of each state in the calculation of vector r_j^*. In this paper, all the states have the same weights, i.e., a uniform probability distribution. Here, $\|\cdot\|_{\epsilon}$ denotes weighted Euclidean norm respect to the weight vector ϵ.

The solution of the least squares minimization problem (47) in the compact form is

$$r_j^* = (\Phi' \Theta \Phi)^{-1} (\Phi' \Theta) J_j^* \tag{48}$$

where $\Theta \in \mathbb{R}_+^{\Omega \times \Omega}$ is the diagonal matrix having the vector $\epsilon \in \mathbb{R}_+^{\Omega}$ along the diagonal. The term $(\Phi' \Theta \Phi)^{-1}$ exists since the feature matrix Φ is full column rank [20] (prime stands for transpose).

For the large state space systems, the calculation of the vector r_j^* for each time slot requires matrix inversion and multiplication of impractical size. Therefore, it is suggested to select a small number of rows of Φ and the corresponding elements of J_j^*, and use least squares on the samples to calculate the approximation $\hat{r}_j^*$ of the parameter vector r_j^*. In particular, we gather $z \in \mathbb{N}$ samples from the state space at each time slot j, where $q < z << \Omega$. The samples are chosen randomly based on a uniform probability distribution.

If we denote by $\hat{\Phi}_j \in \mathbb{R}^{z \times q}$, $\hat{\tilde{J}}_j^* \in \mathbb{R}^z$, the *sampled feature matrix* and the *sampled value function*, respectively, the following regression based unweighted least squares problem then will be solved:

$$\hat{r}_j^* = \arg\min_{r \in \mathbb{R}^q} \{\|\hat{\tilde{J}}_j^* - \hat{\Phi}_j r\|^2 + \beta \|r - \bar{r}\|^2\}, \quad j = 0, 1, \ldots, T-1 \tag{49}$$

where $\bar{r}$ is an initial guess of the solution at each time slot, $\beta \in \mathbb{R}_+$ is positive coefficient. The vector r_j^* is replaced by $\hat{r}_j^*$ since we solve a low dimensional least squares problem. Here, $\|\cdot\|$ denotes norm-2.

If we show by $\hat{\xi}^v(j)$ any arbitrary state chosen randomly at the time slot j, the sampled value function $\hat{\tilde{J}}_j^*(\hat{\xi}^v)$ in (49) is calculated by the following formula at each time slot:

$$\hat{\tilde{J}}_j^*(\hat{\xi}^v) = R(\hat{\xi}^v) + \max_{u(j) \in U} \sum_{\xi^w \in \Xi} p_{\xi^v \xi^w}(u) \phi'(\xi^w) \hat{r}_{j+1}^*$$
$$j = 0, 1, \ldots, T-1 \tag{50}$$

where $\hat{r}_T^*$ is provided in advance. One might consider either a heuristic guess or any approximation method to replace an appropriate value for $\hat{r}_T$. The pseudo-code of the USLS-ADP algorithm shown in Algorithm 1 (note that the current time slot k is assumed to be equal to 0).

The (finite) sequence of the parameter vectors $\hat{r}_j^*$ inherits both the contraction mapping and monotonicity properties of the exact DP algorithms. The mathematical convergence of the sequence $\hat{r}_j^*$ is only applicable over an infinite time horizon, for which the USLS-ADP algorithm converges [19]. Such converge is guaranteed by the aforementioned contraction mapping and monotonicity properties. The USLS-ADP convergence over an infinite time horizon is indicative of the goodness (boundedness) of the calculated solutions, when applied over a finite time horizon.

In the literature, there are well-known Approximate DP algorithms supported by similar arguments, for instance see the Sequential DP Approximation algorithm [7]. Like the proposed USLS-ADP, it is a sample-based algorithms, is used over a finite time horizon, and inherits the contraction mapping and monotonicity properties of the exact DP algorithms.

A. Definition of the Basis Functions for the Resource Allocation Problems

For the USLS-ADP approach, we define the following $q = m + 1$ basis (or feature) functions with the state variable $\xi = (\xi_1, \ldots, \xi_m)'$ as argument:

$$\phi_0(\xi) = 1, \quad \phi_i(\xi) = \xi_i, \quad i = 1, \ldots, m. \tag{51}$$

The related linear approximation is

$$\tilde{J}(\xi, r) = r_0 + \sum_{i=1}^{m} r_i \xi_i. \tag{52}$$

where $r' = (r_0, r_1, \ldots, r_m)$ is the parameter vector (note that r_i is the i-th component of the generic parameter vector r related to ξ_i). The linear relation between the reward function and the number of customers allocated for each price (see (10)) can justify the proposed basis function definition and the associated approximation architecture.

It is worth mentioning that a limited number of well-crafted feature functions can capture the dominant non-linearities of the value function of complex systems, and their linear combination can work well as an approximation architecture, see [8], [20]. In this paper, we do not discuss the construction of the feature functions, even though we note the possibility of their optimal choice within some restricted class by using

gradient and random search algorithms, see [20], [24]. The construction of the basis functions for subspace approximation is an important research issue, and has received considerable attention recently in the ADP literature, see also [25].

Algorithm 1 USLS-ADP Algorithm [19]

• Choose sample size z, time horizon T, and regularized term β.

• Set the parameter vector $r_T \in \mathbb{R}^q$ for the terminal condition, i.e., $\tilde{J}_T(\xi, r) = \phi(\xi) r_T$.

• $j \leftarrow T$

while $j \geq 1$

• $j \leftarrow j - 1$

• Sample the original state space with uniform probability distribution, construct the sampling matrix $\Delta_j \in \mathbb{R}^{z \times \Omega}$ and the sampling set $\hat{\Xi}_j$.

• Compute $\hat{J}_j^*$

$$\hat{J}_j^*(\hat{\xi}^v) = R(\hat{\xi}^v) + \max_{u \in U} \sum_{\xi^w \in \Xi} p_{\hat{\xi}^v \xi^w}(u) \phi(\xi^w) \hat{r}_{j+1}^*,$$
$$\hat{\xi}^v \in \hat{\Xi}_k,\ \xi^w \in \Xi.$$

• Compute $\hat{\Phi}_j = \Delta_j \Phi$.

• Solve the following least squares problem:

$$\hat{r}_j^* = \arg \min_{r_j \in \mathbb{R}^q} \left\{ \|\hat{\Phi}_j r_j - \hat{J}_j^*\|_2^2 + \beta \|r_j - \hat{r}_{j+1}^*\|_2^2 \right\}$$

by the following formula:

$$\hat{r}_j^* = (\hat{\Phi}_j' \hat{\Phi}_j + \beta I)^{-1} (\hat{\Phi}_j' \hat{J}_j^* + \beta \hat{r}_{j+1}^*).$$

• Define the projected value function $\tilde{J}_j^*(\xi^v) = \phi(\xi^v) \hat{r}_j^*$, for any given state $\xi^v \in \Xi$.

end while

B. Reservation Price Algorithm Adaptation

The reservation pricing algorithm can be adapted for resource allocation problems with large state space. In particular, the modifications to be applied to the procedure described in Section V can be summarized as follows:

• The exact DP algorithm has to be replaced with the USLS-ADP algorithm.

• As a consequence, the optimal value function $J_j^{*,x^j}(\cdot)$ is calculated approximately in the feature sub-space. Thus, we compute the parameter vector $\hat{r}_j^*$ and the approximate optimal value function $\tilde{J}_j^{*,x^j}(\xi^v) = \phi(\xi^v) \hat{r}_j^*$, for any given state $\xi^v \in \Xi$.

• This means that the notation of all the value functions has to be updated, e.g., $J_k^{V^h(u)}(\bar{\xi}(k))$ becomes $\tilde{J}^{V^h(u)}(\bar{\xi}(k))$.

• Monte Carlo simulations are used to compute the terms composing $\tilde{J}^{V^h(u)}(\bar{\xi}(k))$ (see (34)).

VIII. Simulation Results

The proposed approach has been evaluated over numerical cases to show its effectiveness. Both exact DP and ADP techniques have been used with the support of on-purpose developed MATLAB programs. The latter is configurable, meaning that they provide the capability of defining the values of the problem to be solved, e.g., λ's and μ's for $i \in \{1, \ldots, m\}$, the number of resources (N) and prices (m), and the time horizon (T). For all the examples we have used the basis functions given by (52), i.e., the number of resources associated to each price c_i multiplied by parameter r_i.

It can be shown that the state space explosion can occur even with relatively small values of m and N. Indeed, increments in such parameters cause an exponential growth in the size of the state space [19]. Hence, the exact DP becomes impractical even for relatively small problem instances. The simulation examples are divided into two main parts. The first part is dedicated to the DP reservations algorithm results, while the second part is for the USLS-ADP algorithm.

Policies computed by DP techniques can be represented by means of lookup tables. In other words, for a given state and time slot j, one can associate the corresponding action calculated by the proposed algorithm. However, such representation can be impractical even for small state and action spaces. Therefore, more compact representations could be used [18]. In this regard, we apply a statistical index showing the frequency distribution of each action at each time slot over the entire state space. It is worth highlighting that such statistical index for policies can also be impractical in cases of large action spaces.

In all the proposed examples, we have used the following state dependent birth and death probabilities:

$$\mu_i(\xi_i) = \mu_{\max}(c_i)(1 - e^{\frac{-\xi_i}{N}})$$
$$\lambda_i(\xi_i) = \lambda_{\min}(c_i) + (\lambda_{\max}(c_i) - \lambda_{\min}(c_i)) e^{\frac{-\xi_i}{N}}. \quad (53)$$

We have assumed that the death (birth) probability increases (decreases) with the number of the allocated resources.

Example 2: Number of prices $m = 3$, number of resources $N = 6$, price $c = [0.9\ 1.0\ 1.1]$, $\lambda_{\max} = [0.55\ 0.5\ 0.3]$, $\lambda_{\min} = [0.3\ 0.2\ 0.1]$, $\mu_{\max} = [0.5\ 0.6\ 0.6]$.

The cardinality of the state space is 84, and the exact DP algorithm can be applied. Here, for the sake of simplicity, we enclosed the prices in bracket, e.g., $c = [c_1\ c_2\ c_3]$.

The following operational scenario has been considered:

• Specified time interval reservation requests at the consecutive time slots $j = 2, 3, 4$ with the duration reported in Table I.

TABLE I
SPECIFIED TIME INTERVAL RESERVATION REQUESTS FOR THE EXAMPLE 2

j	2	3	4
h_1	12	13	14
h_2	16	17	21

• Unspecified time interval reservation request at the time slot $j = 1$ for the time slot $h = 20$.

The frequency distribution of the different actions (normalised versus the state space cardinality and over the finite time horizon $T = 40$) is shown in Fig. 3. In particular, the above-defined operational scenario (with reservation requests) is compared with the case of no reservation requests, where the DP off-line solution calculated at the beginning can be applied for instant resource needs. To analyze the result, one has to start from the terminal stage and move backwards. Since there are neither unspecified nor specified time interval

reservations for the time slots $j > 21$, based on the principle of optimality, the frequency distributions of the reservation and no reservation cases are identical.

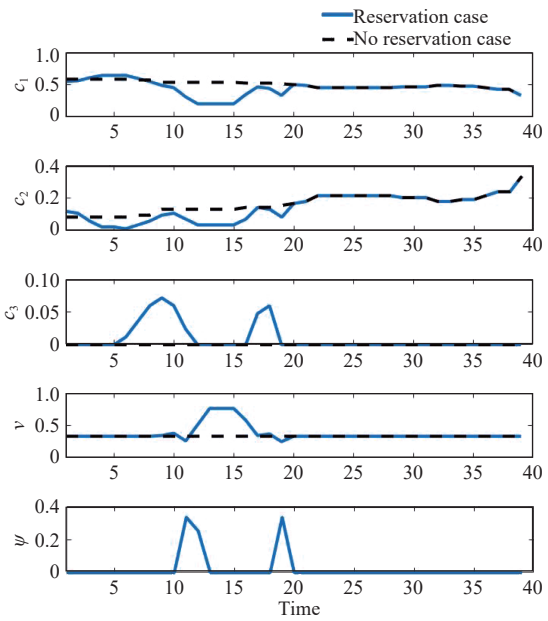

Fig. 3. Comparison of the action frequency distributions for the reservation and no-reservation cases (Example 2).

As shown in the same figure, the frequency distribution of the rejection control ν increases for the intervals with reservations. It is noticed that for the case of no reservation, the control ψ is never applied. Therefore, it is not plotted.

Additionally, the specified time interval reservation vector set x and the unspecified time interval reservation vector set s are depicted in Fig. 4. As shown in the figure, the price c_3 is more likely to be chosen than the others. Moreover, for the case of specified time interval reservations, the algorithm does not propose the price c_2 to the costumer requests; hence, we do not plot the associated plot. In the case of unspecified time interval reservations, the algorithm only proposes the price c_3; hence, we do not plot the other prices.

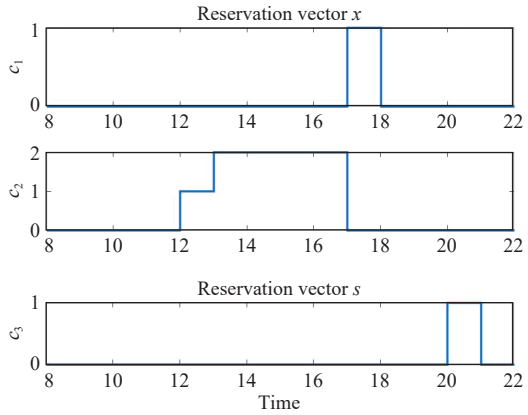

Fig. 4. Specified and unspecified time interval reservation vectors x and s (Example 2).

Finally, it is worth highlighting that the specified time interval reservation request $[h_1, h_2] = [14, 21]$ is rejected by the algorithm.

Example 3: The same resource allocation problem set-up of the Example 2 is considered, but with a more complex operational scenario.

The following operational scenario has been considered:

• Specified time interval reservation requests handled at the time slots $j = 1, 4, 5, 7, \ldots, 12$ with the durations reported in Table II.

TABLE II
SPECIFIED TIME INTERVAL RESERVATION REQUESTS FOR THE EXAMPLE 3

j	1	4	5	7	8	9	10	11	12
h_1	14	14	14	14	14	14	14	14	14
h_2	21	16	16	19	19	19	21	21	18

• Unspecified time interval reservation requests handled at the time slots $j = 2, 3, 6$ for the time slots $h = 12, 22, 25$, respectively.

Thanks to the principle of optimality, the frequency distribution curves of the reservation and no reservation cases are identical for the time slots $j > 25$, see Fig. 5. However, for the time intervals $14 \le j \le 16$, there are differences between such curves. For the interval $14 \le j \le 16$, the rejection policy ν is on its maximum point since the system is fully reserved for the associated time slots.

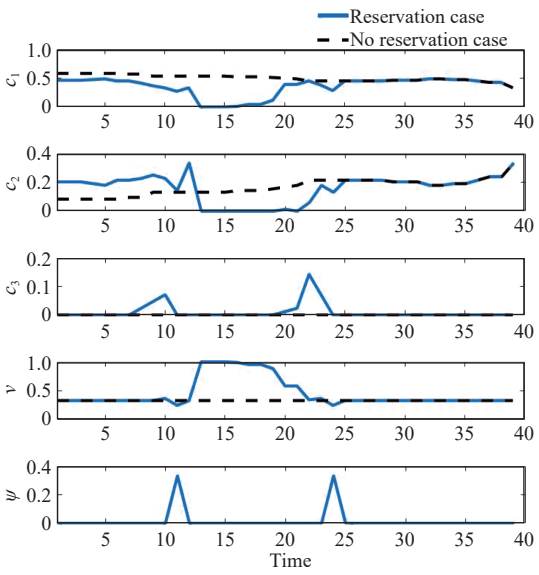

Fig. 5. Comparison of the action frequency distributions for the reservation and no-reservation cases (Example 3).

Furthermore, the vectors x and s are depicted in Fig. 6. Unlike the previous scenario, the price c_2 is proposed for some intervals. The total number of customers corresponding to reservation vectors x and s (regardless of the associated prices) are depicted in Fig. 7, which shows that all the resources are allocated for the time interval $14 \le j \le 16$.

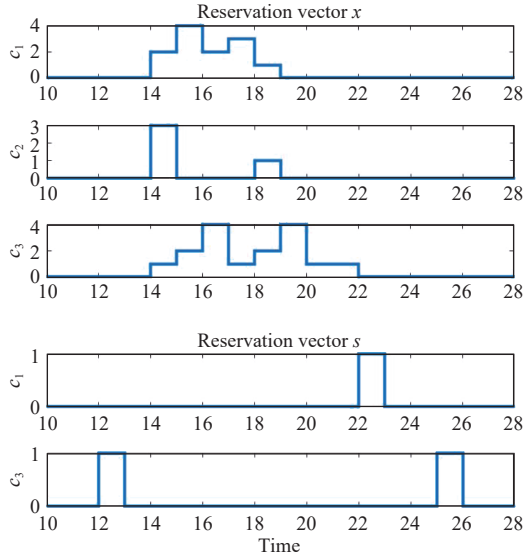

Fig. 6. Specified and unspecified time interval reservation vectors x and s (Example 3).

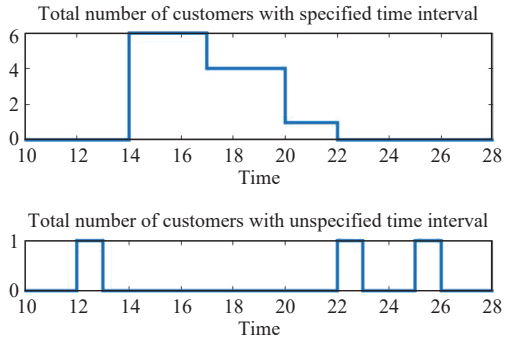

Fig. 7. Total number of customers at different time slots (Example 3, Scenario 2).

Example 4: Number of prices $m = 4$, number of resources $N = 30$, price $c = [0.9\ 1\ 1.1\ 1.2]$, $\lambda_{\max} = [0.55\ 0.5\ 0.3\ 0.2]$, $\lambda_{\min} = [0.3\ 0.25\ 0.1\ 0.08]$, $\mu_{\max} = [0.5\ 0.6\ 0.6\ 0.62]$.

Due to the large state space, the reservation pricing algorithm with the USLS-ADP is employed. In this example, the following operational scenarios have been considered:

1) Only instant resource requests.

2) Specified time interval reservation requests handled at the consecutive time slots $j = 1,2,\ldots,7$ with the duration reported in Table III.

TABLE III
SPECIFIED TIME INTERVAL RESERVATION REQUESTS FOR THE SCENARIO 2 OF THE EXAMPLE 4

j	1	2	3	4	5	6	7
h_1	13	10	12	17	23	15	19
h_2	17	20	14	26	32	17	26

3) Specified time interval reservation requests handled at the consecutive time slots $j = 7,8,10,11$ with the duration reported in Table IV; unspecified time interval reservation requests handled at the time slots $j = 1,\ldots,6,9,12$ for the time slots $h = 4,6,5,11,22,14,26,26$, respectively.

TABLE IV
SPECIFIED TIME INTERVAL RESERVATION REQUESTS FOR THE SCENARIO 3 OF THE EXAMPLE 4

j	7	8	10	11
h_1	12	17	25	18
h_2	15	17	28	20

4) Specified time interval reservation requests handled at the consecutive time slots $j = 2,4,\ldots,7$ with the duration reported in Table V; unspecified time interval reservation requests handled at the time slots $j = 1,3$ for the time slots $h = 25,27$, respectively.

TABLE V
SPECIFIED TIME INTERVAL RESERVATION REQUESTS FOR THE SCENARIO 4 OF THE EXAMPLE 4

j	2	4	5	6	7
h_1	23	28	30	26	32
h_2	26	28	30	29	32

In all these operational scenarios, the terminal value function has been calculated by means of a Temporal Difference based approach over an infinite time horizon. The value of $\|\hat{r}_j^*\|$ of such scenarios is plotted in Fig. 8. Generally speaking, it is noteworthy that the reservation time and its duration affect the value of $\|\hat{r}_j^*\|$. Starting from the terminal stage and moving backwards, it can be noted that all the curves increase in the same way. Such behavior is in line with the definitions of the scenarios, i.e., for the time slots $j \geq 32$ no resource is assigned for future utilization. The curve of the Scenario 4 grows faster than the other ones in the interval $23 \leq j \leq 32$, since such scenario targets all the advance resource requests in that time interval. On the whole, the Scenario 3 exhibits the higher growth rate of $\|\hat{r}_j^*\|$ since the reservation requests span over the entire time slots $1 \leq j \leq 26$. The Scenarios 1, 2, and 4 have the same growth rate for the time slots $j \leq 12$, where they have no allocated resource. However, they have a different offset due to the different allocated resources for $j > 12$. The Scenario 1 has no resource reservation, thus the associated parameter vector has the lowest profile.

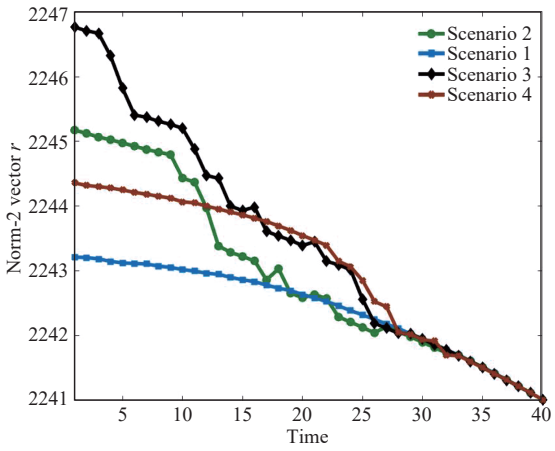

Fig. 8. Norm-2 of the vector $\hat{r}_j^*$ for different scenarios defined in Example 4.

IX. Related Work

In this section, an essential literature review about the most pertinent articles on MDP modeling and relative solutions for resource allocation problems is provided. A homogeneous continuous-time Markov chain is proposed in [26] to model the patient flows for the hospital ward management. The optimization of the matching between the resources and the demands is performed by means of a local search heuristic algorithm. MDPs are employed in [27] for Business Process Management with the goal of making appropriate decisions to allocate the resources by trying to minimize the long-term cost and to improve the performance of the business process execution. A heuristic based Reinforcement Learning approach is adopted as optimization method. In [28], a resource allocation problem for the vehicular cloud computing systems is discussed. Since the objective is the maximization of the total long-term expected reward, the optimization is formulated as an infinite horizon MDP with the state space, action space, reward model, and transition probability distribution of the particular case study. An iteration algorithm is utilized to develop the optimal scheme that computes which action has to be taken in a certain state. MDP based modelling and solution methodology for scheduling patients in a multi-class, multi-resource surgical system is employed in [29]. The proposed model provides a scheduling policy for all surgeries, and minimizes a combination of the lead time between patient request and surgery date, overtime in the operating room, and congestion in the wards. A least square temporal difference ADP approach is to deal with the curse of dimensionality. One of the most important operations in the production of growing-finishing pigs is the marketing of pigs for slaughter. In [30], a sequential marketing decisions at the herd level is considered as a high dimensional infinite horizon MDP. A value iteration ADP algorithm is used to estimate the value function for this infinite time horizon problem. The stochastic behavior of the food banks inventory system has been modelled by using an MDP in [31], which has the advantage of indicating the best way to allocate supplies based on the inventory level of the food bank. Such paper presents a novel transformation of the state space to account for the large distribution quantities observed in practice and shows that the particular underlying stochastic behavior can be approximated by a normal distribution. Similarly to our approach, both stochastic and deterministic aspects are addressed. In [32], a sequential resource allocation problem with an objective function aimed at equitable and effective service for the problem of distributing a scarce resource to meet the customer's demands is carried out. In this work, through a DP framework, the structure of the optimal allocation policy for a given sequence of the customer's demand is characterized as continuous probability distributions. In this regard, by using the identified optimal structure, a heuristic base allocation policy for the instances with discrete demand distribution has been proposed.

In some other works, resource allocation problems are treated as multi-agent systems. In [33], for instance, the dynamic of the agents is considered as second order differential equations, while they communicate over weight-balanced and strongly connected digraphs. The optimization problem is formulated as a constrained convex objective function. The effectiveness of the method, however, is evaluated for a small number of agents, only. An alternative approach can be found in [34], where the distribution of a common resource between two sources of time varying demand is carried out to develop the time-efficient methods for minimizing delays at severely congested airports. In this work, the problem is formulated as a DP optimization and the objective is based on the second moments of the stochastic queue lengths. It is shown that for sufficiently high volumes of the demand, optimal values can be well approximated by the quadratic functions of the system state. Again, a heuristic based approach is applied as ADP method. A comparison between Monte Carlo tree search and rolling horizon optimisation approaches is carried out in [35] for two challenging dynamic resource allocation problems: the wild fire management and the queuing network control. Even though this work shows interesting results, the reported techniques are suitable just for the specific applications considered. Another example of resource allocation strategies can be found in [36], where the problems of budget allocations of non profit organizations on geographically distinct areas is tackled. The proposed solution consists in formulating the overall resource allocation problem as a two-stage stochastic mixed integer programming problem. A heuristic-based approach is finally used to simplify the original formulation.

An interesting variant to the solution of (stochastic) resource allocation problems is represented by the MPC, especially suited when the dynamics of the systems is considered to be variable over time (time-variant processes). In [37], for instance, it is shown how the stochastic resource allocation problem can be addressed by suitably modifying the MDP and the optimal control problem and using MPC to allocate resources over time. In particular, a new class of algorithms for the stochastic unreliable resource allocation problem is proposed, when there is a chance that the task will not be completed by that resource. However, a well-defined and accurate prediction model is a priori needed for an effective strategic allocation control. Similarly, in [38], the solution for the stochastic resource allocation problem makes use of MPC integrated with machine learning and Markov chain model. The theory is based on a three layer lambda architecture and particularly tailored to the case study of a dispatch decision problem from an energy distribution utility.

As a general remark, it is noted that our paper provides a sufficiently comprehensive modeling framework, which is not limited to a specific application. Moreover, the optimization algorithms for price policy calculation exploit the most advanced ADP techniques to address the scalability issues of real-world applications, instead of resorting to heuristic or example driven methods. To the best of the authors' knowledge, the current literature on this topic do not have these features.

Admittedly, one of main difficulties of applying ADP based approaches is the choice of proper basis functions, which is a

current area of research (see Section VII-A). Moreover, it is assumed to know the system stochastic mechanisms and the related probability distributions. If this is not feasible, one can resort to computer simulators (generating samples according to such probability distribution) or Reinforcement Learning based approaches [8], [12].

X. Conclusions

Resource reservations in resource allocation problems have been modelled as a general-purpose MDP based framework. Stochastic DP based approaches have been proposed to compute proper pricing policies, and show how Bellman's principle of optimality can play a role in the implementation of the resulting pricing reservation algorithms. However, the resulting framework, which also includes an overbooking business strategy, becomes computationally intractable in case of realistic resource allocation problems. As a consequence, ADP techniques based on linear function approximations have been employed to solve such scalability issues. In particular, the novel USLS-ADP algorithm has been applied. Examples addressing both specified and unspecified time interval reservation requests have been shown, solved, and analyzed to verify the soundness of the proposed approach.

As for future work, we plan to apply the proposed framework to relevant business applications, such as flight ticket booking, urban parking management, and smart energy management systems. This implies defining the probability distributions of the underlying stochastic processes. In case of their unavailability, it is possible to resort to computer simulators (generating samples according to such probability distribution) or reinforcement learning based approaches.

References

[1] D. Bertsimas, S. Gupta, and G. Lulli, "Dynamic resource allocation: A flexible and tractable modeling framework," *European J. Operational Research*, vol. 236, pp. 14–26, 2014.

[2] S. M. Kandil, H. E. Z. Farag, M. F. Shaaban, and M. Z. El-Sharafy, "A combined resource allocation framework for PEVs charging stations, renewable energy resources and distributed energy storage systems," *Energy*, vol. 143, pp. 961–972, 2018.

[3] J. Acimovic and S. C. Graves, "Making better fulfillment decisions on the fly in an online retail environment," *Manufacturing & Service Operations Management*, vol. 17, no. 1, pp. 1–18, 2014.

[4] T. Liu, B. Tian, Y. Ai, Y. Zou, and F. Y. Wang, "Parallel reinforcement learning-based energy efficiency improvement for a cyber-physical system," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 2, pp. 617–626, 2020.

[5] L. Jiang, H. Huang, and Z. Ding, "Path planning for intelligent robots based on deep Q-learning with experience replay and heuristic knowledge," *IEEE/CAA J. Autom. Sinica*, 2019.

[6] A. Pietrabissa, F. D. Priscoli, A. D. Giorgio, A. Giuseppi, M. Panfili, and V. Suraci, "An approximate dynamic programming approach to resource management in multi-cloud scenarios," *Int. J. Control*, vol. 90, pp. 492–503, 2017.

[7] D. P. Bertsekas, "Dynamic Programming and Optimal Control, Vol. I" Athena Scientific, Belmont, Massachusetts, USA, 2017. [Online]. Available: http://www.athenasc.com/dpbook.html

[8] D. P. Bertsekas, "Dynamic Programming and Optimal Control, Vol. II" Athena Scientific, Belmont, Massachusetts, USA, 2012. [Online]. Available: http://www.athenasc.com/dpbook.html

[9] C. Quan and F. Ren, "Weighted high-order hidden Markov models for compound emotions recognition in text," *Information Sciences*, vol. 329, pp. 581–596, 2016.

[10] V. M. Baskaran, Y. C. Chang, J. Loo, K. Wong, and M. Gan, "Dominant speaker detection in multipoint video communication using Markov chain with non-linear weights and dynamic transition window," *Information Sciences*, vol. 464, pp. 344–362, 2018.

[11] A. Pietrabissa, M. Castrucci, and A. Palo, "An MDP approach to faulttolerant routing," *European J. Control*, vol. 18, no. 4, pp. 334–347, 2012.

[12] D. P. Bertsekas, "Feature-based aggregation and deep reinforcement learning: A survey and some new implementations," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 1, pp. 1–31, 2019.

[13] D. P. Bertsekas, "Dynamic programming and suboptimal control: A survey from ADP to MPC," *European J. Control*, vol. 11, no. 4, pp. 310–334, 2005.

[14] Z. Zhang and D. Zhao, "Clique-based cooperative multiagent reinforcement learning using factor graphs," *IEEE/CAA J. Autom. Sinica*, vol. 1, no. 3, pp. 248–256, 2014.

[15] M. Kumar, K. Rajagopal, S. N. Balakrishnan, and N. T. Nguyen, "Reinforcement learning based controller synthesis for flexible aircraft wings," *IEEE/CAA J. Autom. Sinica*, vol. 1, no. 4, pp. 435–448, 2014.

[16] A. Forootani, M. Tipaldi, M. G. Zarch, D. Liuzza, and L. Glielmo, "Modelling and solving resource allocation problems via a dynamic programming approach," *Int. J. Control*, 2019.

[17] A. Forootani, D. Liuzza, M. Tipaldi, L. Glielmo, "Allocating resources via price management systems: a dynamic programming-based approach," *Int. Journal of Control*, 2019.

[18] D. P. Bertsekas, "Temporal difference methods for general projected equations," *IEEE Trans. Automatic Control*, vol. 56, no. 9, pp. 2128–2139, 2011.

[19] A. Forootani, R. Iervolino, and M. Tipaldi, "Applying unweighted least squares based techniques to stochastic dynamic programming: Theory and application," *IET Control Theory & Applications*, vol. 13, no. 15, pp. 2387–2398, 2019.

[20] D. Bertsekas, "Approximate policy iteration: A survey and some new methods," *J. Control Theory and Applications*, vol. 9, no. 3, pp. 310–335, 2011.

[21] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to Probability*, Athena Scientific Optimization and Computation Series, 2nd. ed., Belmont, Massachusetts, USA, 2008.

[22] D. P. de Farias and B. Van Roy, "The linear programming approach to approximate dynamic programming," *Operations Research*, vol. 51, no. 6, pp. 850–865, 2003.

[23] H. Yu and D. P. Bertsekas, "Convergence results for some temporal difference methods based on least squares," *IEEE Trans. Automatic Control*, vol. 54, no. 7, pp. 1515–1531, 2009.

[24] L. Busoniu, D. Ernst, B. De Schutter, and R. Babuska, "Cross-entropy optimization of control policies with adaptive basis functions," *IEEE Trans. Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 41, no. 1, pp. 196–209, 2011.

[25] D. P. Bertsekas and H. Yu, "Projected equation methods for approximate solution of large linear systems," *J. Computational and Applied Mathematics*, vol. 227, pp. 27–50, 2009.

[26] A. R. Andersen, B. F. Nielson, and L. B. Reinhardt, "Optimization of hospital ward resources with patient relocation using Markov chain modeling," *European J. Operational Research*, vol. 260, no. 3, pp. 1152–1163, 2017.

[27] Z. Huang, W. M. Van Der Aalst, Xu. Lu, and H. Duan, "Reinforcement learning based resource allocation in business process management," *Data & Knowledge Engineering*, vol. 70, no. 1, pp. 127–145, 2011.

[28] K. Zheng, H. Meng, P. Chatzimisios, L. Lei, and X. Shen, "An SMDP based resource allocation in vehicular cloud computing systems," *IEEE Trans. Industrial Electronics*, vol. 62, no. 12, pp. 7920–7928, 2015.

[29] D. Astaraky and J. Patrick, "A simulation based approximate dynamic programming approach to multi-class multi resource surgical scheduling," *European J. Operational Research*, vol. 245, no. 1, pp. 309–319, 2015.

[30] R. Pourmoayed and L. R. Nielsen, "An approximate dynamic programming approach for sequential pig marketing decisions at herd level," *European J. Operational Research*, vol. 276, no. 3, pp. 1056–1070, 2019.

[31] S. Fianu and L. B. Davis, "A Markov decision process model for equitable distribution of supplies under uncertainty," *European J. Operational Research*, vol. 264, pp. 1101–1115, 2018.

[32] R. W. Lien, S. M. R. Iravani, and K. R. smilowitz, "Sequential resource allocation for nonprofit operations," *Operations Research*, vol. 62, no. 2, pp. 301–317, 2014.

[33] Z. Deng, "Distributed algorithm design for resource allocation problems of second-order multi-agent systems over weight-balanced digraphs," *IEEE Trans. Systems*, *Man*, *and Cybernetics*, 2019.

[34] R. Shone, K. Glazebrook, and K. G. Zografos, "Resource allocation in congested queueing systems with time-varying demand: An application to airport operations," *European J. Operational Research*, vol. 276, no. 2, pp. 566–581, 2019.

[35] D. Bertsimas, J. D. Grifth, V. Gupta, M. J. Kochenderfer, and V. V. Misic, "Comparison of Monte Carlo tree search and rolling horizon optimization for large-scale dynamic resource allocation problems," *European J. Operational Research*, vol. 263, no. 2, pp. 664–678, 2017.

[36] A. Bayram, S. Solak, and M. Johnson, "Stochastic models for strategic resource allocation in non profit foreclosed housing acquisitions," *European J. Operational Research*, vol. 233, no. 1, pp. 246–262, 2014.

[37] D. Castanon, J. M. Wohletz, "Model predictive control for stochastic resource allocation," *IEEE Trans. Automatic Control*, vol. 54, no. 8, pp. 1739–1750, 2009.

[38] M. P. Dal Pont, R. S. Ferreira, W. W. Teixeira, D. M. Lima, and J. E. Normey-Rico, "MPC with machine learning applied to resource allocation problem using Lambda architecture," *IFAC-Papers OnLine*, vol. 52, no. 1, pp. 550–555, 2019.

Ali Forootani (M'19) received the M.Sc. degree in electrical engineering and automatic control system from Power and Water University of Technology, Iran, in 2011, and the Ph.D. degree in automatic control system from Department of Engineering, University of Sannio, Italy, in 2019. From 2011 to 2012, he worked on the monitoring of high voltage circuit breakers and wind turbine control system at Niroo Research Institute, Iran. He worked as a control and instrumentation Engineer at the Ministry of Power and Energy (Water and Sewage khuzestan Engineering Company) from 2012 to 2015. He worked as a Research Fellow at the Measurement and Instrumentation Laboratory University of Sannio, as well as Post Doc. Research Fellow at the University of Salerno from Dec. 2018 to Jan. 2020. Currently he is doing research at Hamilton Institute, Maynooth University, Ireland. His current research interests include Markov decision processes, approximate dynamic programming, reinforcement learning in optimal control, and learning in network control systems. He is a Member of IEEE Control System Society and Reviewer of several important journals.

Raffaele Iervolino (M'19) received the laurea degree cum laude in aerospace engineering from the University of Naples, Italy, in 1996, where he also obtained the Ph.D. degree in electronic and computer science engineering in 2002. Since 2003 he is an Assistant Professor of automatic control at the University of Naples. From 2005 he is also a Adjoint Professor of automatic control with the Department of Electrical Engineering and Information Technology at the same University. He is Member of IEEE Control System Society. His research interests include piecewise affine systems, opinion dynamics and consensus in social networks, and human telemetry systems.

Massimo Tipaldi received the master degree in computer science engineering and the Ph.D. degree in information technology from the University of Sannio, Italy, in 1998 and 2017, respectively. He possesses more than 20 years of industrial experience in the managerial and technical coordination of ESA/ASI/CNES space project (satellite systems, experimental equipment for the International Space Station, and ground segments). He holds 1 patent and has co-authored more than 40 papers published in proceedings of international conferences or international archival journals. His research interests include space systems engineering, critical SW systems, reinforcement learning, approximate dynamic programming, stochastic systems, and advanced system control techniques.

Joshua Neilson is a Ph.D. candidate in information technology at the University of Sannio, Italy, where he studies optical coatings for gravitational wave interferometers. He received the master degree in physics from California State University, USA, in 2017. He has been a Member of the LIGO Scientific Collaboration since 2016, has co-authored several peer-reviewed papers and more than 40 LIGO collaboration papers.