

RESEARCH ARTICLE

## Performance comparison of approximate dynamic programming techniques for dynamic stochastic scheduling

Yasin Göçgün\*

*Department of Industrial Engineering, IstinYE University, Istanbul, Turkey*  
*yasin.gocgun@istinye.edu.tr*

---

### ARTICLE INFO

#### Article History:

*Received 08 June 2020*

*Accepted 01 March 2021*

*Available 09 May 2021*

#### Keywords:

*Dynamic stochastic scheduling*

*Markov decision processes*

*Approximate dynamic programming*

#### AMS Classification 2010:

*90-08; 60Jxx*

---

### ABSTRACT

This paper focuses on the performance comparison of several approximate dynamic programming (ADP) techniques. In particular, we evaluate three ADP techniques through a class of dynamic stochastic scheduling problems: Lagrangian-based ADP, linear programming-based ADP, and direct search-based ADP. We uniquely implement the direct search-based ADP through basis functions that differ from those used in the relevant literature. The class of scheduling problems has the property that jobs arriving dynamically and stochastically must be scheduled to days in advance. Numerical results reveal that the direct search-based ADP outperforms others in the majority of problem sets generated.



---

## 1. Introduction

Approximate dynamic programming (ADP) is a method to solve large-scale Markov decision processes (MDPs), which are used to model systems that evolve stochastically over time. The term approximate refers to the fact that the solution obtained by the underlying ADP technique is an approximate to the optimal solution. ADPs have been used to solve problems arising in diverse fields such as healthcare, manufacturing, transportation, and revenue management.

In the last few decades, various ADP techniques have been proposed to approximately solve computationally intractable MDPs. The state-of-the-art ADP techniques include the Lagrangian-based ADP ([1], [2]), the linear programming-based ADP ([3], [4]), and the direct search-based ADP techniques ([5], [6]). However, the performances of those techniques have not been evaluated in the literature.

To this end, we evaluate the performances of the aforementioned ADP techniques through a class of dynamic stochastic scheduling problems. These

problems have the following main features: 1) Jobs arrive dynamically and stochastically at the system over time; 2) Arriving jobs from different types must be scheduled to future time slots such as days. These problems are termed as dynamic stochastic advanced scheduling problems (DSASPs) and arise in various fields such as manufacturing, healthcare, and transportation. We perform a comparison analysis, considering diverse scenarios obtained by different levels of crucial problem parameters. In particular, we approximately solve various problem sets generated for a class of DSASPs introduced in [3] using the aforementioned ADP techniques. The way we implement the direct search-based ADP is unique in that we use new basis functions for value function approximation.

The rest of the paper is structured as follows. Section 2 discusses the relevant literature. In Section 3, we introduce a class of dynamic stochastic advanced scheduling problems, describe the ADP techniques to be compared, and present our computational work. Section 4 includes concluding remarks.

---

\*Corresponding author

## 2. Literature review

We review the literature on both approximate dynamic programming (ADP) and dynamic stochastic advanced scheduling. A summary of work done on ADP is given first.

### 2.1. Literature review on approximate dynamic programming

There are many studies focusing approximate dynamic programming (ADP). Powell [7] provides a good review on ADP-based techniques. We briefly review prominent work on ADP below.

Farias and Roy [8] addressed the curse of dimensionality of large-sized stochastic control problems by developing linear programming (LP)-based ADP for solving such problems. In the heart of their approach, a linear combination of basis functions is fitted to cost-to-go function. The authors developed error bounds that ensure performance guarantees. In another study, Farias and Van Roy [9] improved linear programming approach to ADP through the development of constraint sampling. They showed that a subset of constraints can be chosen independently of total number of constraints the problem contains under certain conditions.

Maxwell et al. [5] proposed ADP-based algorithms for ambulance redeployment. In particular, the authors introduced direct search based ADP for solving the underlying problem. Owing to the computationally intensive nature of direct search, they utilized a “post-decision state dynamic programming formulation of ambulance redeployment”.

Shechter et al. [10] studied an optimal search problem where the location of a target is only known probabilistically. The authors aimed to minimize the probability of having a failed search and considered the “unconstrained search” and the “constrained search”. They developed ADP approaches for larger instances of their problem. Numerical results showed that ADP-based algorithms perform well. In a follow-up study, Gocgun [11] worked on a class of optimal search problems that contain a target and an obstacle. The author provided Markov decision process (MDP) formulations of these problems and proposed a direct search-based ADP for obtaining approximate solutions.

Gocgun and Ghate [12] studied a class of dynamic resource allocation problems where “multiple renewable resources must be dynamically allocated to different types of jobs arriving randomly”. The objective is to select “which jobs to service in

each time-period so as to maximize total infinite-horizon discounted expected profit.” The authors developed a Lagrangian relaxation-based ADP method for obtaining approximate solutions to those problems. In a follow-up study, Gocgun and Ghate [1] proposed an ADP approach based on Lagrangian relaxation for dynamic stochastic scheduling problems. Their computational results demonstrated that the ADP approach outperforms myopic decision rules.

Yin et al. [13] studied a class of metro train scheduling problems, considering performance metrics such as time delay of passengers and operational costs. They proposed a stochastic programming model for this problem and approximately solved it through an ADP-based algorithm.

Wang et al. [14] introduced ADP-based methods through iterated Bellman inequalities. Their methods solve linear and semidefinite programs and provide a bound on optimal value as well as a reasonably good suboptimal policy.

Li and Womer [15] studied a class of project scheduling problems that contain resource constraints and task durations that are uncertain. Differing from the existing research, the authors found a dynamic and adaptive policy through ADP-based algorithms. Specifically, they developed a hybrid ADP framework that makes use of the rollout policy as well as a lookup table approach.

Nozhati et al. [16] developed a framework for recovery management. Their approach utilizes ADP and heuristics for determining recovery actions. Their approach efficiently manages multi-state systems following disasters.

Yang et al. [17] proposed ADP-based algorithms for optimization problems with nonlinear constraints. In particular, they introduced a policy-iteration algorithm to solve the underlying problem, and validated their control method through the simulation of an interconnected plant.

Kanj et al. [18] employed ADP for a problem faced in a ride-hailing system that consists of a fleet of autonomous electric vehicles. Through ADP, the authors developed dispatch strategies to determine, for instance, which car is the most appropriate for a particular trip. Their work showed that the problem contains monotone value functions.

Ou et al. [19] studied a class of gantry scheduling problems where the material transfer is handled by gantries. The authors introduced a method that makes use of reinforcement learning and ADP. Numerical results showed that the proposed

method outperforms a standard Q-learning algorithm.

## 2.2. Literature review on dynamic stochastic advanced scheduling

One prominent feature of dynamic stochastic advanced scheduling problems (DSASPs) is that they are formulated as Markov decision processes (MDPs). We review research on DSASPs below.

Patrick et al. [3] introduced an MDP formulation of DSASPs where patients from different types are scheduled to future days. Due to intractable state and action spaces, the authors employed an ADP approach; specifically they developed an LP-based ADP to provide approximate solutions. In a follow-up study, Saure et al. [4] studied a DSASP faced in radiation therapy units. The authors provided an MDP formulation of the underlying problem and solved it using an ADP method that is based on linear programming.

Gocgun and Puterman [20] worked on an appointment booking problem faced in chemotherapy settings. Differing from the similar problems, it has the property that patients have target dates along with tolerance limits. The authors proposed an LP-based ADP for acquiring an optimal solution. In a follow-up study, Gocgun [6] worked on a DSASP faced in chemotherapy settings and allows for cancellation of jobs. The author employed a direct search-based ADP for solving larger instances of the underlying problem.

Akhavizadegan et al. [21] addressed appointment scheduling in a nuclear medical center, considering patient choice and different no-show rates. The authors formulated the problem as an MDP and compared the optimal solution with heuristic decision rules. Wang and Fung [22] studied a class of dynamic appointment scheduling problems considering patient preferences and choices. The authors developed a column generation-based approximation algorithm to solve these problems. Lu et al. [23] worked on a class of dynamic appointment scheduling problems taking into account “wait-dependent abandonment”. They formulated these problems as MDPs and investigated the properties of the optimal policy theoretically.

In a recent work, Saure et al. [24] studied a DSASP where service times are stochastic. The authors put forth theoretical results for the deterministic case with “multi-class, multi-priority” jobs, and then developed methods for the stochastic case.

## 2.3. Contribution

Table 1 indicates research work in which a comparative analysis was performed using any of the LP-based ADP (LP-A), the direct search-based ADP (DS-A), and the Lagrangian-based ADP (LGR-A) techniques. To the best of our knowledge, the relevant literature does not contain any work that deals with a comparative analysis of all the three state-of-the-art ADP techniques.

**Table 1.** The list of work in which any of LP-A, DS-A, and LGR-A was developed. “S-based A” refers to Simulation-based ADP.

Study	Proposed ADP	Comparison
[5]	DS-A	against S-based A
[10]	DS-A	against heuristics
[11]	DS-A	against heuristics
[12]	LGR-A	against myopic
[1]	LGR-A	against myopic
[3]	LP-A	against myopic
[4]	LP-A	against myopic
[20]	LP-A	against myopic
[6]	DS-A	against myopic
[2]	LGR-A	against myopic

In this research, we evaluate the performance of three state-of-the-art ADP techniques, employing them for solving a class of DSASPs. In particular, the Lagrangian-based ADP, the LP-based ADP, and the direct-search based ADP were used to solve the DSASP introduced in [3]. Our contribution is twofold: 1) We employ the direct search-based ADP through basis functions that are different as compared to those used in the literature, 2) We close a gap in the literature, addressing the question of which of those techniques perform the best in an important class of dynamic scheduling problems.

The features of the DSASP we studied are given next.

## 3. Dynamic stochastic advanced scheduling

The dynamic stochastic advanced scheduling problem introduced in [3] has the following features.

- Heterogeneous job types are considered.
- Arrivals of jobs to the system are random. In addition, arrivals across job types are independent.

- Jobs arriving at the system must be scheduled to a day within a booking horizon. Rejecting (or outsourcing or serving through overtime) jobs is allowed.
- There is a deadline for jobs of each type. Scheduling a job to a day after its deadline results in delay cost.
- Rejecting jobs results in a penalty cost.
- The goal is to make decisions of scheduling and rejecting arriving jobs so as to “minimize the total discounted expected cost over an infinite horizon” ([20]).

### 3.1. The Markov decision process model

The following notations are used in the mathematical model of the aforementioned problem.

- $I$ : the number of job types
- $N$ : the length of the booking horizon
- $x_n, n = 1, \dots, N$ : number of jobs that are already scheduled to day  $n$
- $u_i, i = 1, \dots, I$ : number of type- $i$  jobs waiting to be scheduled
- $y_{in}, i = 1, \dots, I, n = 1, \dots, N$ : number of type- $i$  jobs to be scheduled to day  $n$
- $z_i, i = 1, \dots, I$ : number of type- $i$  jobs that are rejected
- $C_1$ : daily capacity
- $C_2$ : upper bound on number of jobs rejected each day
- $p(u'_i)$ : probability that  $u'_i$  jobs of type- $i$  arrive on a given day
- $D^i$ : a deadline associated with type- $i$  job
- $C^i(n, D^i)$ : delay cost of scheduling a type- $i$  job on day  $n$
- $r(i)$ : rejection cost of a type- $i$  job
- $F^i, i = 1, \dots, I$ : unit delay cost for a type- $i$  job
- $D$ : the set of all possible demand vectors

The Markov decision process (MDP) model of the aforementioned problem is provided next (see [3] for an equivalent formulation).

**State Space:**  $s = (x, u) = (x_n, u_i), i = 1, \dots, I$  and  $n = 1, \dots, N$ . The state of the system consists of the number of jobs that are already scheduled to each day in a booking horizon, and the number of jobs of each type waiting to be scheduled.

**The Action Set:**  $(y, z) = (y_{in}, z_i), i = 1, \dots, I$  and  $n = 1, \dots, N$ . The action to be made at a given state is to decide the number of jobs of each type to be scheduled to each day of the booking horizon, and the number of jobs of each type that are rejected. Note that  $z_i$  does not have the day index, as it represents the number of jobs of type- $i$  that will not be scheduled to any day of

the booking horizon and hence are rejected. Any action must satisfy certain constraints, which are provided below ([3]).

$$x_n + \sum_{i=1}^I y_{in} \leq C_1, \quad n = 1, \dots, N, \quad (1)$$

$$\sum_{i=1}^I z_i \leq C_2, \quad (2)$$

$$\sum_{n=1}^N y_{in} + z_i \leq u_i, \quad i = 1, \dots, I. \quad (3)$$

Constraint 1 ensures that the sum of the number of jobs that are already scheduled to day  $n$  and total number of jobs to be scheduled to day  $n$  does not exceed the daily capacity. Constraint 2 guarantees that total number of jobs rejected is bounded by  $C_2$ . Finally, Constraint 3 ensures that the sum of the total number of type- $i$  jobs to be scheduled and the number of type- $i$  jobs rejected cannot be greater than the number of type- $i$  jobs waiting to be scheduled.

**Transition Probabilities:** Stochasticity in the system arises only due to the number of new arrivals of jobs from each type. Hence, once an action is chosen at a given state  $(x_1, x_2, \dots, x_N, u_1, u_2, \dots, u_I)$ , the system switches to the following state with probability  $\prod_{i=1}^I p(u'_i)$  due to the assumption of independent arrivals:

$$(x_2 + \sum_{i=1}^I y_{i2}, x_3 + \sum_{i=1}^I y_{i3}, \dots, x_N + \sum_{i=1}^I y_{iN}, 0, u'_1, u'_2, \dots, u'_I).$$

Here, for instance,  $x_2 + \sum_{i=1}^I y_{i2}$  represents  $x'_1$ .

**Costs:** The immediate cost of choosing an action at a given state consists of total delay cost and total rejection cost. It is mathematically expressed as follows.

$$c(y, z) = \sum_{i=1}^I \sum_{n=1}^N C^i(n, D^i) y_{in} + \sum_{i=1}^I r(i) z_i.$$

$$C^i(n, D^i) = \max(n - D^i, 0) \times F^i, \quad n = 1, \dots, N. \quad (4)$$

**Bellman's Equations:** The cost-to-go function of a given state is given by

$$v(x, u) = \min_{(y, z)} \left\{ c(y, z) + \lambda \sum_{u' \in D} (u') v(x', u') \right\}. \quad (5)$$

Owing to extremely large number of states and actions, the underlying MDP model is computationally intractable. The three approximate dynamic programming techniques are briefly described next.

### 3.2. Approximate dynamic programming techniques

Due to curse of dimensionality, Bellman equations given in Eqn. 5 cannot be solved. The fundamental theme behind approximate dynamic programming (ADP) is to approximate the value function (i.e., cost-to-go function) through a combination of basis functions, thereby eliminating the computational intractability.

ADPs are mainly categorized as mathematical programming (MP)-based ADP, simulation-based ADP, and direct search-based ADP. MP-based ADPs transform the underlying MDP model into the “equivalent linear programming (LP) version of Bellman equations. Approximate value function is then used to avoid intractability” ([20]). Examples of MP-based ADPs are linear programming (LP)-based ADP and Lagrangian-based ADP. Simulation-based ADP techniques, however, simulate “the evolution of the system over a number of initial states in order to tune the parameters” ([6]), thereby finding an approximate solution to the Bellman’s equations. Simulation models such as reinforcement learning and statistical sampling are used to estimate value functions. On the other hand, ADP based on direct search tackles an optimization problem where the decision variables are tuning parameters, and the goal is to minimize “the expected cost of the policy induced by the corresponding parameter vector” ([6]). The optimization problem is solved through direct search.

As stated earlier, in ADP, basis functions that possess certain important features of the system state are used to approximate the value function. One example of utilizing basis functions is linear approximation, which is given by

$$V(s) \approx \sum_{k=1}^K r_k \Phi_k(s),$$

where “ $r_k$  for  $k = 1, \dots, K$  are tuning parameters and  $\Phi_k(s)$  for  $k = 1, \dots, K$  are basis functions” ([11]). The approximation parameters are tuned iteratively to acquire an ADP policy after the approximation of the value function is performed. In this context, ADP approaches aim to find the optimal parameter vector through which a certain performance metric is minimized ([11]).

The parameter tuning phase enables us to have the approximate value of a given state. We then retrieve the ADP policy through the computation of a decision vector for any given state.

We briefly describe the three approximate dynamic programming techniques, without delving

into all mathematical details. (refer to [3], [1] and [6] for technical details of these methods).

#### 3.2.1. Linear programming-based ADP

The LP approximation is provided below (see [3] for the complete steps of the LP-based ADP).

“For a discounted infinite-horizon MDP (where the objective function is in minimization form as in (5) and  $\alpha(\vec{s})$  are positive numbers indexed by states  $\vec{s} \in S$ ), the equivalent LP formulation is given” ([20]):

$$\begin{aligned} & \max \sum_{\vec{s} \in S} \alpha(\vec{s})v(\vec{s}) \\ & s.t. \ c(\vec{s}, \vec{a}) + \\ & \lambda \sum_{\vec{s}' \in S} p(\vec{s}'|\vec{s}, \vec{a})v(\vec{s}') \geq v(\vec{s}), \ \forall \vec{s} \in S, \vec{a} \in A_{\vec{s}}. \end{aligned} \tag{6}$$

Using an affine approximation, the value function can be approximated as:

$$\tilde{v}(\vec{x}, \vec{u}) = W_0 + \sum_{n=1}^N V_n x_n + \sum_{i=1}^I W_i u_i. \tag{7}$$

The LP formulation of our MDP model is then:

$$\begin{aligned} & \max_v \sum_{(\vec{x}, \vec{u}) \in S} \alpha(\vec{x}, \vec{u})v(\vec{x}, \vec{u}) \\ & s.t. \ c(\vec{y}, \vec{z}) + \\ & \lambda \sum_{d \in D} p(d)v(x_2 + \sum_i y_{i2}, \dots, x_N + \sum_i y_{iN}, 0, u'_i) \\ & \geq v(\vec{x}, \vec{u}), \ \forall (\vec{x}, \vec{u}) \in S, \forall (\vec{y}, \vec{z}) \in A_{(\vec{x}, \vec{u})}. \end{aligned} \tag{8}$$

We substitute (7) into (6) and obtain the following LP after rearranging terms ([20]):

$$\begin{aligned} & \max_{\vec{V}, \vec{W}} \ W_0 + \sum_{n=1}^N E_{\alpha}(X_n)V_n + \sum_{i=1}^I E_{\alpha}(U_i)W_i \\ & s.t. \ (1 - \lambda)W_0 + \\ & \sum_{n=1}^N V_n(x_n - \lambda x_{n+1} - \lambda \sum_{i=1}^I y_{i(n+1)}) + \\ & \sum_{i=1}^I W_i(u_i - \lambda E_{\alpha}(U_i)) \leq c(\vec{y}, \vec{z}), \ \forall (\vec{x}, \vec{u}) \in S, \\ & \forall (\vec{y}, \vec{z}) \in A_{(\vec{x}, \vec{u})}, \\ & V_n \geq 0, \ n = 1, \dots, N, \\ & W_i \geq 0, \ i = 1, \dots, I. \end{aligned} \tag{9}$$

As the above LP “still has a very large number of constraints” ([20]), its dual is solved through column generation (see [3]).

### 3.2.2. Lagrangian relaxation-based ADP

The Lagrangian approach is similar to the LP-based in that it transforms the underlying MDP and tackles the equivalent LP formulation of the MDP through the problem decomposition obtained by Lagrange multipliers. As the resulting LP is still intractable, a hybrid Lagrangian relaxation - LP approach is employed to tackle intractability. In particular, the Lagrangian value functions are approximated through affine functions. The resulting approximate LP is solved using a column generation method. ([25], [1])

### 3.2.3. Direct-search based ADP

As part of the direct search-based ADP, we tune approximation parameters using direct search with the goal of finding good policies. To be more specific, an optimization problem where feasible  $r$ 's constitute the variables and the goal is to minimize "the expected cost of the policy induced by the corresponding parameter vector" ([6]) is solved by direct search. As a result, we have the following optimization problem ([11]):

$$\min_{r \in R^N} \sum_{t=0}^{\infty} c(s_t, \pi_r(s_t)), \quad (10)$$

where " $s_t$  is the state at stage  $t$  of the system,  $\pi_r$  is the policy obtained by the parameter vector  $r$ ,  $\pi_r(s_t)$  is the action dictated by the policy  $\pi_r$  in the state at stage  $t$ , and  $c(s_t, \pi_r(s_t))$  is immediate cost incurred at step  $t$  as a result of choosing  $\pi_r(s_t)$ ." [11]

We use the following basis functions during the implementation of the direct-search based ADP.

$$\Phi_1(s) = C_1 - \sum_{n=1}^N \sum_{i=1}^I (x_{in} + y_{in}), \quad (11)$$

$$\Phi_2(s) = -\left(\sum_{i=1}^I z_i\right).$$

The first basis function represents available capacity (see [6] for a somewhat similar basis function), whereas the second one allows us to consider different values of  $z_i$  for the underlying optimization problem.

Because of having two basis functions, two tuning parameters are used, which are  $r_1$  and  $r_2$ . We let  $r_1$  and  $r_2$  range from 1 to 5 in increments of 1, and 0 to 40 in increments of 2, respectively. For each problem instance, the combination of  $(r_1, r_2)$  that yields the best value is used for computing average cost values.

### 3.3. Numerical experiments

Data generation was performed by taking into account the way data is generated in the literature ([1]). Number of types was set to 5 and 10. Arrival probabilities of jobs are assumed to follow Poisson with a parameter DU (1,5) (DU means discrete uniform). Discount factor was set to 0.9 and 0.99. Resource availability was set to 10 and 20. Two levels were considered for booking horizon: 7 and 14. As a result, we have 16 scenarios for the comparison analysis.

As the arrival process is random, we estimate the discounted expected cost accrued by any of the three ADP techniques by averaging the total discounted cost through simulation. Simulation run length was set to 50, and number of replications was set to 20, which means that the total discounted cost is averaged over 20 independent simulations. For each problem set, we ran 10 problem instances.

#### 3.3.1. Results

We provide results in tables 2 and 3. For each problem set determined by the combination of  $I, C_1$ , and  $N$ , columns 2 to 4 of each table give the average discounted cost values over 10 independent problem instances obtained for the Lagrangian-based, the LP-based, and the direct search-based ADP, respectively. The last column of each table gives the percentage difference between the best and next best techniques. The bolded percentage difference values correspond to problem sets where the Lagrangian-based ADP outperforms others whereas other values correspond to problem sets where the direct search-based ADP outperforms others. When the discount factor ( $\lambda$ ) has a high level (i.e., 0.99); the Lagrangian-based ADP turns out to be the best approach in 5 out of 8 problem sets, whereas the direct search-based ADP outperforms others in two problem sets. When the discount factor was set to a low level (i.e., 0.9), the direct search-based ADP outperforms others in all problem sets. (Paired  $t$ -tests revealed that the respective percentage differences were statistically significant at the 0.05 level.)

**Table 2.** Results for  $\lambda = 0.99$ .

$(I, C_1, N)$	LGR-A	LP-A	DS-A	Per. d.
(5,5,7)	15379	20208	16991	<b>9.5</b>
(5,5,14)	15273	20127	15763	<b>3.1</b>
(5,10,7)	6546	10380	7345	<b>10.9</b>
(5,10,14)	6831	9556	6676	2.3
(10,10,7)	31110	41513	33120	<b>6.1</b>
(10,10,14)	31256	40179	31035	0.7
(10,20,7)	12310	21844	13062	<b>5.7</b>
(10,20,14)	12627	16746	11174	11.5

**Table 3.** Results for  $\lambda = 0.90$ .

$(I, C_1, N)$	LGR-A	LP-A	DS-A	Per. d.
(5,5,7)	3758	3785	3346	11
(5,5,14)	3758	3197	2762	13.6
(5,10,7)	1708	1685	1327	21.2
(5,10,14)	1708	1172	1022	12.8
(10,10,7)	7276	7514	6451	11.3
(10,10,14)	7276	5978	5185	13.3
(10,20,7)	2923	2827	2351	16.8
(10,20,14)	2923	1541	1461	5.2

#### 4. Conclusions

In this paper, we aimed to close a gap in the literature by comparing the performances of the state-of-the-art approximate dynamic programming (ADP) techniques through a class of dynamic stochastic advanced scheduling problems (DSASPs). These problems are modeled as Markov decision process and their large instances are approximately solved via ADP techniques. We solved a class of these problems using three ADP approaches: 1) Lagrangian-based ADP, 2) Linear programming-based ADP, and 3) direct search-based ADP, which we uniquely implemented through new basis functions.


Our numerical experiments reveal that the direct search-based ADP outperforms others in 10 out of 16 problem sets. On the other hand, the Lagrangian-based ADP outperforms others in 5 out of 16 problem sets. Future research may focus on the performance comparison of such techniques through variants of DSASPs that include extensions such as cancellations of jobs, multiple resources, and overbooking.

#### References

- [1] Gocgun, Y., & Ghate, A. (2012). Lagrangian relaxation and constraint generation for allocation and advance scheduling. *Computers & Operations Research*, 39, 2323-2336.
- [2] Parizi, M. S., & Ghate, A. (2016). Multi-class, multi-resource advance scheduling with no-shows, cancellations and overbooking. *Computers & Operations Research*, 67, 90-101.
- [3] Patrick, J., Puterman, M. L., & Queyranne, M. (2008). Dynamic multi-priority patient scheduling for a diagnostic resource. *Operations Research*, 56, 1507-1525.
- [4] Saure, A., Patrick, J., Tyldesley, S., & Puterman, M. L. (2012). Dynamic multi-appointment patient scheduling for radiation therapy. *European Journal of Operational Research*, 223, 573-584.
- [5] Maxwell, M.S., Henderson, S. G., & Topaloglu, H. (2013). Tuning approximate dynamic programming policies for ambulance redeployment via direct search. *Stochastic Systems*, 3(2), 322-361.
- [6] Gocgun, Y. (2018). Dynamic scheduling with cancellations: An application to chemotherapy appointment booking. *An International Journal of Optimization and Control: Theories and Applications*, 8, 2, 161-169.
- [7] Powell, W. B. (2007). *Approximate dynamic programming: solving the curses of dimensionality*, Hoboken, New Jersey, USA: John Wiley and Sons.
- [8] De Farias, D.P. & Roy, B.V. (2003). The linear programming approach to approximate dynamic programming. *Operations Research*, 51, 850-865.
- [9] De Farias, D. P. & Roy, B. V. (2004). On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3), 462-478.
- [10] Shechter, S. M., Ghassemi, F., Gocgun, Y., & Puterman, M. L. (2015). Technical note – trading off quick versus slow actions in optimal search. *Operations Research*, 63(2), 353-362.
- [11] Gocgun, Y. (2019). Approximate dynamic programming for optimal search with an obstacle. *Selcuk University Journal of Engineering, Science, and Technology*, 7, 80-95.
- [12] Gocgun, Y., & Ghate, A. (2010). A Lagrangian approach to dynamic resource allocation. *Proceedings of the Winter Simulation Conference*, 3330-3338.
- [13] Yin, J., Tang, T., Yang, L., Gao, Z., & Ran, B. (2016). Energy-efficient metro train rescheduling with uncertain time-variant passenger demands: an approximate dynamic programming approach. *Transportation Research, Part B*, 91, 178-210.

- [14] Wang, Y., O'Donoghue, B., & Boyd, S. (2015). Approximate dynamic programming via iterated bellman inequalities. *International Journal of Robust and Nonlinear Control*, 25, 1472–1496.
- [15] Li, H., & Womer, N. K. (2015). Solving stochastic resource-constrained project scheduling problems by closed-loop approximate dynamic programming. *European Journal of Operational Research*, 246, 20-33 2015.
- [16] Nozhati, S., Sarkale, Y., Ellingwood, B., Chong, E. K. P., & Mahmoud, H. (2019). Near-optimal planning using approximate dynamic programming to enhance post-hazard community resilience management. *Reliability Engineering and System Safety*, 181, 116-126.
- [17] Yang, X., He, H., & Zhong, X. (2019). Approximate dynamic programming for nonlinear-constrained optimizations. *IEEE Transactions on Cybernetics*.
- [18] Al-Kanj, L., Nascimento, J., & Powell, W. B. (2020). Approximate dynamic programming for planning a ride-sharing system using autonomous fleets of electric vehicles. *European Journal of Operational Research*, 284, 1088-1106.
- [19] Ou, X., Chang, Q., & Chakraborty, N. (2021). A method integrating Q-Learning with approximate dynamic programming for gantry work cell scheduling. *IEEE Transactions on Automation Science and Engineering*, 18, 1, 85-93.
- [20] Gocgun, Y., & Puterman, M. L. (2014). Dynamic scheduling with due dates and time windows: an application to chemotherapy patient appointment booking. *Health Care Management Science*, 17, 60-76.
- [21] Akhavizadegan, F., Ansarifar, J., & Jolai, F. (2017). A novel approach to determine a tactical and operational decision for dynamic appointment scheduling at nuclear medical center. *Computers & Operations Research*, 78, 267–277.
- [22] Wang, J., Fung, R.Y., & Chan, H.K. (2015). Dynamic appointment scheduling with patient preferences and choices. *Industrial Management & Data Systems*, 115(4), 700-717.
- [23] Lu, Y., Xie, X., & Jiang, Z. (2018). Dynamic appointment scheduling with wait-dependent abandonment. *European Journal of Operational Research*, 265(3), 75-984.
- [24] Saure, A., Begen, M.A. & Patrick, J. (2020). Dynamic multi-priority, multi-class patient scheduling with stochastic service times. *European Journal of Operational Research*, 280(1), 254–265.
- [25] Gocgun, Y. (2010). *Approximate dynamic programming for dynamic stochastic resource allocation with applications to healthcare*. PhD Thesis. The University of Washington.

**Yasin Göçgün** received his B.S. degree and M.S. degree from the Industrial Engineering Department at Bilkent University in 2003 and 2005, respectively. After completing his doctoral studies in the Industrial and Systems Engineering Department at the University of Washington in 2010, Dr. Göçgün worked as a postdoctoral fellow in the Sauder School of Business at the University of British Columbia between 2010 and 2012. After his first post-doc, Dr. Göçgün carried out another post-doc study in the Mechanical and Industrial Engineering Department at the University of Toronto between 2012 and 2014. He has been working as an assistant professor in the Industrial Engineering Department at Istinye University. His research interests primarily focus on dynamic stochastic optimization, operations research in healthcare, Markov decision processes, approximate dynamic programming, and scheduling.

 <https://orcid.org/0000-0003-3005-7596>

