Operations Research Perspectives 8 (2021) 100204

Contents lists available at ScienceDirect

Operations Research Perspectives

journal homepage: www.elsevier.com/locate/orp

A review of approximate dynamic programming applications within military operations research

M. Rempel [a,*], J. Cai [b]

[a] Centre for Operational Research and Analysis, Defence Research and Development Canada, 101 Colonel By Dr., K1A 0K2, Ottawa, Canada
[b] Canadian Joint Operations Command, 1600 Star Top Road, K1B 3W6, Ottawa, Canada

ARTICLE INFO

Keywords:
Sequential decision problem
Markov decision process
Approximate dynamic programming
Reinforcement learning
Military

ABSTRACT

Sequences of decisions that occur under uncertainty arise in a variety of settings, including transportation, communication networks, finance, defence, etc. The classic approach to find an optimal decision policy for a sequential decision problem is dynamic programming; however its usefulness is limited due to the curse of dimensionality and the curse of modelling, and thus many real-world applications require an alternative approach. Within operations research, over the last 25 years the use of Approximate Dynamic Programming (ADP), known as reinforcement learning in many disciplines, to solve these types of problems has increased in popularity. These efforts have resulted in the successful deployment of ADP-generated decision policies for driver scheduling in the trucking industry, locomotive planning and management, and managing high-value spare parts in manufacturing. In this article we present the first review of applications of ADP within a defence context, specifically focusing on those which provide decision support to military or civilian leadership. This article's main contributions are twofold. First, we review 18 decision support applications, spanning the spectrum of force development, generation, and employment, that use an ADP-based strategy and for each highlight how its ADP algorithm was designed, evaluated, and the results achieved. Second, based on the trends and gaps identified we discuss five topics relevant to applying ADP to decision support problems within defence: the classes of problems studied; best practices to evaluate ADP-generated policies; advantages of designing policies that are incremental versus complete overhauls when compared to currently practiced policies; the robustness of policies as scenarios change, such as a shift from high to low intensity conflict; and sequential decision problems not yet studied within defence that may benefit from ADP.

1. Introduction

Many decisions are not made in isolation—decisions are made; new information, which was previously uncertain, is observed; given this new information, further decisions are made; more new information arrives; and so on. These types of decisions are aptly described as *sequential decision problems*, *sequential decision making under uncertainty*, or *multistage decision problems* and are characterized by decisions having an impact on future rewards received or costs incurred, the feasibility of future decisions, and in some cases the exogenous events that occur between decisions [1–3]. In essence, "today's decisions impact on tomorrow's and tomorrow's on the next day's" [2, p. 1], and if the relationship between decisions is not accounted for, then the outcomes achieved may be neither efficient nor effective.

It has been known since the 1950s that such sequential decisions may be modelled as a Markov Decision Process (MDP), which consists of five components: a set of candidate actions; rewards that are received as a result of selecting an action; the epochs at which decisions are made; the state, which is the information required to select an action, determine the rewards, and inform how the system evolves; and transition probabilities that define how the system transitions from one state to the next [4]. Given a MDP, the objective is then to find a decision policy—"*a rule (or function) that determines a decision given the information available*" [3, p. 221], also referred to as a contingency plan, plan, or strategy [2, p. 22]—that makes decisions which result in the system performing optimally with respect to a given criterion. The classic approach to finding an optimal decision policy is to solve Bellman's optimality equation via Dynamic Programming (DP) [5]. Within a defence context, DP has been applied to determine decision policies for a variety of sequential decision problems, including fleet maintenance and repair [6], scheduling basic training [7], selecting research and development projects [8], stay-or-leave decisions for military personnel [9], and dispatching medical evacuation assets [10].

* Corresponding author.
E-mail addresses: mark.rempel@forces.gc.ca (M. Rempel), juliacai@cmail.carleton.ca (J. Cai).

https://doi.org/10.1016/j.orp.2021.100204
Received 26 May 2021; Received in revised form 26 September 2021; Accepted 30 September 2021
Available online 14 October 2021

Although DP provides an elegant framework to solve sequential decision problems, its limited usefulness in many real-world applications has been long acknowledged. This is due to the curse of dimensionality [5]—"the extraordinarily rapid growth in the difficulty of problems as the number of variables (or dimensions) increases" [11]—and the curse of modelling, which is the need for an explicit model of how the system transitions from one state to the next [12]. While today's computers can solve sequential decision problems with millions of states [13], many problems remain too large to be solved efficiently via classic DP methods. In addition, it is often the case that the transition probabilities between states are simply not known. Sequential decision problems with these characteristics permeate throughout defence, spanning the spectrum of force development, generation, and employment. For example:

- within force development, decisions regarding capability investments, which may number in the hundreds, typically occur at fixed times during a business planning cycle and repeat on a yearly basis. Decision makers must consider both the short and long-term impact of the selected investments, as well as the investments not selected, while accounting for uncertainty surrounding future military obligations, changes in coalition and adversaries' capabilities, defence specific inflation, etc.;
- within force generation, decisions regarding how many commissioned and non-commissioned members to recruit in order to meet requirements across the spectrum of military occupations, while respecting the nation's authorized strength and accounting for various uncertainties including yearly retirements, promotions, attrition, etc.; and
- within force employment, decisions regarding which individuals to load onto helicopters during a mass evacuation operation, such as a major maritime disaster, while accounting for uncertainties including changes in weather, the individuals' health, helicopter breakdowns, etc.

As a result of these challenges, in these types of problems it is often not possible to find an optimal decision policy, and alternative approaches that focus on finding a good or near-optimal policy are required. The first approach, based on functional approximation, was suggested by Bellman and Dreyfus [14], with additional approaches being developed throughout the following decades across a variety of fields, including operations research, control theory, and computer science—see Powell [15] for a detailed discussion and list of relevant references. In addition, the field of mathematical programming, in particular stochastic programming, has developed sophisticated algorithms to solve problems with high-dimensional decision and state vectors, often seen in real-world sequential decision problems [16].

Within operations research, these approaches have been developed under a variety of names; in particular, neuro-dynamic programming, adaptive dynamic programming, and Approximate Dynamic Programming (ADP). The popularity of these approaches has grown over the last 25 years, as depicted in Fig. 1, with 2286 articles being published between 1995 and 9 April 2021 and the yearly publication rate growing from a single article to nearly 250 per year. More recently, the term ADP—"a method of making intelligent decisions within a simulation" [17, p. 205] where "the resulting policies are not optimal, so the research challenge is to show that we can obtain [high-quality decision policies] that are robust under different scenarios" [18, p. 3]—has become the more commonly used term [3]. (Authors have recently started to use the label reinforcement learning as well, as evidenced by the recently published book entitled *Reinforcement learning and Optimal Control* [19], and a forthcoming book entitled *Reinforcement Learning and Stochastic Optimization: A unified framework for stochastic decisions* [20].) Of note, ADP-generated decision policies have been successfully deployed into industry, including policies to schedule drivers in the trucking industry [21–23], plan and manage locomotives [24, 25], and manage high-value spare parts within manufacturing [26].

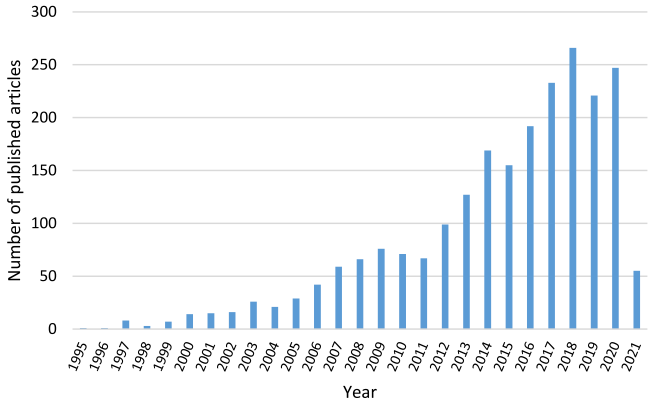

Fig. 1. Number of ADP related articles published per year between 1995 and 9 April 2021.
Source: Web of Science using the search pattern ''approximate dynamic programming'' OR ''adaptive dynamic programming'' OR ''neuro-dynamic programming'' in the title, abstract, and keywords.

In this article we present the first review of applications of ADP set within a defence context. In particular, we focus on peer-reviewed literature within the field of military operations research; that is "[t]he application of quantitative analytic techniques to inform military [or civilian] decision making" [27]. This article's main contributions are twofold. First, we review 18 decision support applications, spanning the spectrum of force development, generation, and employment, that use an ADP-based strategy and for each highlight how its ADP algorithm was designed, evaluated, and the results achieved. Second, based on the trends and gaps identified we discuss five topics relevant to applying ADP to decision support problems in defence: the classes of problems studied; best practices to evaluate ADP-generated policies; advantages of designing policies that are incremental versus complete overhauls when compared to currently practiced policies; the robustness of policies as scenarios change, such as a shift from high to low intensity in a conflict; and we suggest additional sequential decision problems within defence that may benefit from ADP-generated policies.

The remainder of this article is organized as follows. Section 2 provides relevant background information. Section 3 presents the methodology used to conduct this review. Section 4 and Section 5 provide the main body of the review. Section 4 reviews the 18 identified applications of ADP for decision support within defence, and Section 5 presents five topics relevant to applying ADP within a defence context. Finally, concluding remarks are given in Section 6.

2. Background

This section provides background information on MDPs, DP, and ADP. While an exhaustive discussion of each topic is beyond the scope of this work, each section provides sufficient background to support the remaining sections of this article. For the interested reader, Puterman [2] provides an in-depth discussion on MDPs and DP, and Powell [3] provides a comprehensive introduction to ADP from an operations research perspective. In addition, Bertsekas and Tsitsiklis [12] discuss ADP from a controls perspective and Sutton and Barto [13] from a computer science viewpoint.

2.1. Markov decision process

A MDP is a model of sequential decision making under uncertainty, and consists of five components: *decision epochs*, *states*, *actions*, *rewards*, and *transition probabilities* [2, Ch. 2]. These components are briefly described as follows.

- **Decision epoch**: A decision epoch is a point at time t at which a decision is made, where $\mathcal{T}$ is the set of all decision epochs. This set may be a continuum or discrete. When a continuum, decisions may be made continuously, at random points when events occur, or at times chosen by a decision maker. In this case, the model is labelled as *continuous-time*. In the discrete case, decisions are made at all decision epochs and the model is labelled as *discrete time*. In addition, the set $\mathcal{T}$ may be finite or infinite, with the model correspondingly labelled as *finite horizon* or *infinite horizon* respectively.
- **State**: The state of a system $S_t \in \mathcal{S}$, where $\mathcal{S}$ is the set of all possible states, may be defined as the minimally dimensioned function of history that is necessary and sufficient to select an action, determine the rewards or costs associated with the decision, and determine the transition probabilities to the next state [3, p. 179]. The state S_t is also referred to as the *pre-decision state*.
- **Actions**: An action $a_t \in \mathcal{A}_t(S_t)$, where $\mathcal{A}_t(S_t)$ is the set of all possible actions available in state S_t at time t, is an option selected by the decision maker that changes the state of the system such that it is in a new state at $t+1$.
- **Rewards**: A reward is received, or a cost is paid, as a result of a decision maker selecting action a_t while in state S_t. This is defined as a real-valued function $C(S_t, a_t), \forall S_t \in \mathcal{S}, a_t \in \mathcal{A}_t(S_t)$, and is known as the *contribution function*.
- **Transition probabilities**: A transition probability function is a non-negative function $p(s'|S_t, a_t)$ that denotes the probability that given the state S_t and the decision maker's selected action a_t, the system is in state s' at time $t+1$. It is usually assumed that $\sum_{s' \in \mathcal{S}} p(s'|S_t, a_t) = 1$.

Given a MDP, a decision maker's goal is to find a decision policy, also referred to as a contingency plan, plan, or strategy [2, p. 22], that makes a sequence of decisions which results in the system performing optimally with respect to a given criterion. This is known as a *Markov decision problem*. For example, if the criterion is to maximize the expected total discounted contribution over a finite horizon, then this may be stated as [2]

$$\max_{\pi \in \Pi} \mathbb{E}^{\pi} \left(\sum_{t=0}^{T} \gamma^t C(S_t, A_t^{\pi}(S_t)) \right), \tag{1}$$

where T is the final decision epoch, γ is a discount factor, and $A_t^{\pi}(S_t)$ is a decision policy that determines the action selected for a given state. The objective is then to find the best decision policy—"*a rule (or function) that determines a decision given the available information in state* S_t" [3, p. 221]—from the family of decision policies $(A_t^{\pi}(S_t))_{\pi \in \Pi}$. The expectation operator $\mathbb{E}^{\pi}$ implies that the policy affects exogenous events that arise between decision epochs. For example, within the context of a ballistic missile scenario, the decision to launch an interceptor against an incoming missile may influence the adversary's decision to launch further missiles. If the policy does not affect exogenous events, then the operator is written as $\mathbb{E}(\cdot)$. It should also be noted that other objective functions may be used, such as maximizing the average contribution [2, p. 332], minimizing a risk measure [28], or a robust objective [29, p. 114].

2.2. Dynamic programming

In 1957, Bellman [5] showed that for a finite horizon MDP an optimal decision policy can be found by recursively computing Bellman's optimality equation,

$$V_t(S_t) = \max_{a_t \in \mathcal{A}_t(S_t)} \left(C(S_t, a_t) + \gamma \sum_{s' \in \mathcal{S}} p(s'|S_t, a_t) V_{t+1}(s') \right), \tag{2}$$

where the value $V_t(S_t)$ of being in state S_t is the value of taking the optimal decision, resulting in an immediate reward from the contribution function and the expected future value $V_{t+1}(S_{t+1})$. Eq. (2) is referred as the *standard form* of Bellman's equation [3, p. 60]. The resulting optimal decision for each state is then given as

$$A_t^{\pi}(S_t) = \arg\max_{a_t \in \mathcal{A}_t(S_t)} \left(C(S_t, a_t) + \gamma \sum_{s' \in \mathcal{S}} p(s'|S_t, a_t) V_{t+1}(s') \right). \tag{3}$$

For infinite horizon MDPs, the subscript t is dropped as these types of problems are often studied in the steady-state, i.e., $V(s) = \lim_{t \to \infty} V_t(S_t)$ [3, p. 67].

DP is a mathematical optimization approach to solving discrete time Markov decision problems. Regarding finite horizon MDPs, *backwards induction* provides an exact solution [2, p. 92], *value iteration* and *policy iteration* algorithms are employed to solve infinite horizon problems, and *linear programming* may be used for either. Thus, DP should be though of as "a collection of algorithms that can be used to compute optimal policies given a perfect model of the environment as a Markov decision process" [13, p. 73].

2.3. Approximate dynamic programming

In some applications, such as many of those studied in the field of operations research, a decision maker selects a vector of decisions $x_t \in \mathcal{X}_t$, where $\mathcal{X}_t$ represents a feasible region (defined by a set of constraints that depend on S_t, i.e., $\mathcal{X}_t = \mathcal{X}(S_t)$), rather than a single action from a small set of possible actions. Given this, and that the transition probabilities $p(S_{t+1}|S_t, x_t)$ are often not known, Eq. (2) may be restated in *expectation form* [30, p. 240],

$$V_t(S_t) = \max_{x_t \in \mathcal{X}_t} \left(C(S_t, x_t) + \gamma \mathbb{E}(V_{t+1}(S_{t+1})|S_t) \right), \tag{4}$$

where $S_{t+1} = S^M(S_t, x_t, W_{t+1})$. The function $S^M(\cdot)$ is the transition function that returns the next state S_{t+1}, W_{t+1} is exogenous information that arrives after a decision is made at time t and before $t+1$ as depicted in Fig. 2, and the expectation operator replaces the summation over probabilities in Eq. (2) and is over the random variable W_{t+1}. The decision policy is then given as

$$X_t^{\pi}(S_t) = \arg\max_{x_t \in \mathcal{X}_t} \left(C(S_t, x_t) + \gamma \mathbb{E}(V_{t+1}(S_{t+1})|S_t) \right). \tag{5}$$

In many real-world applications, computing an optimal policy via Eq. (4) is not feasible due to the three curses of dimensionality [3, pp. 3–6]: first, the state space $\mathcal{S}$ may be large resulting in a prohibitively large number of times Eq. (4) must be computed; second, for each state S_t, the expectation over W_{t+1} must be computed; and third, the expectation must be computed for each decision x_t. ADP aims to overcome these limitations through employing the concept of the *post-decision state variable* [3, p. 129–139], and using an approximation of the value function. The tradeoff is that the resulting policies are often not optimal; rather, the goal of ADP is to seek high-quality policies that are robust under different conditions.

Similar to DP, ADP should be thought of as a framework rather than a single algorithm. In the remainder of this subsection, we present an overview of key concepts of the ADP framework within the context of a finite horizon MDP as it makes the presentation more explicit. We also assume the action to be taken at time t is represented by a vector of decisions $x_t \in \mathcal{X}_t$, and thus a decision policy is given as $X_t^{\pi}(S_t)$ as opposed to $A_t^{\pi}(S_t)$. First, we discuss how ADP overcomes the curses of dimensionality. Next, we summarize four classes of decision policies, i.e., implementations of Eq. (5). We follow this by discussing the algorithmic strategies employed to search for a good policy, where the choice of algorithm depends on the policy class selected. Lastly, we briefly show how ADP-generated policies are evaluated.

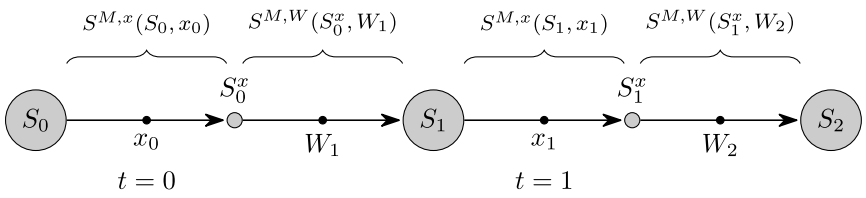

Fig. 2. Example of a sequence of decisions as a MDP with the post-decision state variable. At each decision epoch t a vector of decisions x_t is selected via a decision policy $X_t^\pi(S_t)$.

2.3.1. Addressing the curses of dimensionality

Post-decision state variable. The post-decision state variable S_t^x represents the state of a system after a decision x_t is made, but before new exogenous information W_{t+1} arrives as depicted in Fig. 2. Using this concept, the transition function $S^M(\cdot)$ may be broken down into two steps,

$$S_t^x = S^{M,x}(S_t, x_t), \tag{6}$$

$$S_{t+1} = S^{M,W}(S_t^x, W_{t+1}), \tag{7}$$

resulting in S_t also being labelled as the *pre-decision state variable*. It follows that the optimality equation around the post-decision state is given as [3, p. 138][1]

$$V_{t-1}^x(S_{t-1}^x) = \mathbb{E}\left(\max_{x_t \in \mathcal{X}_t}\left(C(S_t, x_t) + \gamma V_t^x(S_t^x) | S_{t-1}^x\right)\right). \tag{8}$$

The advantage of Eq. (8) over Eq. (4) is that the expectation operator is outside the `max` operator and does not need to be computed for each decision. In addition, Eq. (8) requires solving a series of deterministic optimization problems—a significant advantage over Eq. (4).

Approximating the value function. Eq. (8) assumes $V_t^x(S_t^x)$ is a lookup table with a one-to-one relationship between value and state. While this is an effective approach is some instances, it is a limitation in others. In order to overcome this limitation, ADP-based strategies often approximate the value function around the post-decision state variable $\overline{V}_t^x(S_t^x) \approx V_t^x(S_t^x)$. There are a range of methods to approximate functions [33]. Strategies to approximate the value function (or the decision policy) may be grouped into three strategies [3, p. 223]:

- **Lookup table**: A lookup table returns a discrete value for each state. An example of a lookup table is one based on hierarchical aggregation [3, pp. 290–304];
- **Parametric representation**: A parametric representation is an analytical function, designed by an analyst, that involves a vector of tunable parameters θ. Parametric representations may be characterized as having either a linear or non-linear architecture; however, both are based on basis functions $\phi_f(S_t^x)$, $f \in \mathcal{F}$, where $\mathcal{F}$ is a set of features based on information from the post-decision state variable [3, p. 305]; and
- **Nonparametric representation**: A nonparametric representation is an approach that builds local approximations based on observations. Examples of nonparametric representations are Neural Network (NN)s, kernel regression, k-nearest neighbour, etc. [3, pp. 316–324].

2.3.2. Decision policies

There are four classes of decision policies [3, Ch. 6]: myopic, and its extension known as myopic Cost Function Approximation (CFA); Policy Function Approximation (PFA); Value Function Approximation (VFA); and a Direct Lookahead Approximation (DLA).[2] In addition, hybrid policies may be created by mixing policies from two or more of these categories. Table 1 lists the policy classes, an example policy for each, and a description of the policy class.

These policy classes may be grouped into two categories: *policy search* and *lookahead approximations*, as depicted in Fig. 3 [34]. The policy search category includes PFAs and myopic CFAs; that is, those policies that are characterized by a low-dimensional vector θ that must be tuned and do not explicitly model the downstream impact of a decision. The lookahead approximation category includes DLAs and VFAs. While these approximations also include tunable parameters, these types of policies explicitly model the impact of today's decisions on the future.

2.3.3. Algorithms

The choice of an algorithmic strategy to search for a good decision policy is dependent on the class of decision policy itself as depicted in Fig. 3. As an in-depth discussion of each algorithmic strategy is beyond the scope of this work, for a deeper appreciation of these algorithms see previously cited works by Powell and the references provided below.

Policy function approximation: A PFA is characterized by a vector θ of tunable parameters, which may range from one parameter to thousands, and thus searching for a good decision policy amounts to searching for the best vector θ. This may be tackled by a variety of algorithms, generally described as *stochastic search*, with individual algorithms being categorized as being either derivative-based or derivative-free [20, Ch. 12]. Derivative-based methods tend to employ a stochastic gradient algorithm where the gradient is either based on a numerical derivative, such as in simultaneous perturbation stochastic approximation [35,36], or an exact derivative; in situations where access to the derivative is not possible, such as when a policy is being evaluated using a complex simulation or obtaining observations are expensive in terms of time or money, derivative-free methods, such as genetic algorithms, may be used [35].

Myopic cost function approximation: A myopic CFA is in essence an optimization model. When using a non-parameterized approximation, the decision function may be solved via an appropriate mathematical programming method. When using a parameterized approximation, the parameter vector θ may be tuned via stochastic search while a solver is used to determine a decision for a given parameter vector.

Direct lookahead approximation: When decisions are vector valued a DLA may be modelled as a mathematical program—deterministic in the case of point forecasts, and stochastic when point estimates are replaced by sample realizations. In the latter situation, two-stage or multistage stochastic programming are core approaches to solving direct lookahead policies [16].

Value function approximation: When using this type of approximation, searching for a good policy is akin to tuning the approximation's parameter vector θ. Two common strategies employed are Approximate Value Iteration (AVI) and Approximate Policy Iteration (API). AVI performs one iteration of policy evaluation and uses this information to conduct policy improvement, looping over these two

[1] The discount factor γ is not included in this equation in Powell [3, p. 138], however it is included in other sources, e.g., Mes and Rivera [31] and McKenna et al. [32].

[2] Note that many authors use the term 'cost function approximation' or 'cost-to-go' to refer to a value function approximation [15, p. 328].

Table 1
Summary of decision policies. See [3, Ch. 6] for a complete description.

Class	Example policy and class description
Myopic	$X^\pi(S_t) = \underset{x_t \in \mathcal{X}_t}{\operatorname{argmax}} C(S_t, x_t)$ Makes decisions based on near-term rewards without regard to long-term impacts. An extension of this policy is known as CFA [29, p. 120], which may include a parameter vector θ, changing or adding constraints, or adding a correction term.
PFA	$X^\pi(S_t\|\theta) = \theta_0 + \theta_1 \phi_1(S_t)$ Returns a decision x_t directly based on the state S_t without solving an optimization problem. May take the form of a lookup table, e.g., if military airlift is not available, then use contracted airlift; a parametric model; or a nonparametric model.
VFA	$X_t^\pi(S_t\|\theta) = \underset{x_t \in \mathcal{X}_t}{\operatorname{argmax}} (C(S_t, x_t) + \gamma \sum_{f \in F} \theta_f \phi_f(S_t^x))$ Explicitly accounts for the downstream impact of a decision through approximating its value around the post-decision state variable. Approximation may take the form of a lookup table, parametric, or nonparametric approximation.
DLA	$X^\pi(S_t\|\theta) = \underset{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}}{\operatorname{argmax}} \sum_{t'=t}^{t+H} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$ Makes a decision at $t = 0$ by optimizing decisions over a finite horizon from t to $t + H$. When the number of possible actions and outcomes over a short time horizon are small, an approach based on tree search, using either complete enumeration, Monte Carlo sampling of outcomes, or employing a roll-out heuristic to evaluate what may happen once reaching a state, is feasible [3, pp. 225–227]. When the problem involves a vector of decisions, a deterministic or stochastic DLA based on mathematical programming may be used (deterministic example shown). In the deterministic case, point forecasts of future exogenous information available at time t are used to design the policy, such as [29, pp. 121–122]. In the stochastic case, point estimates are replaced by sample realizations.

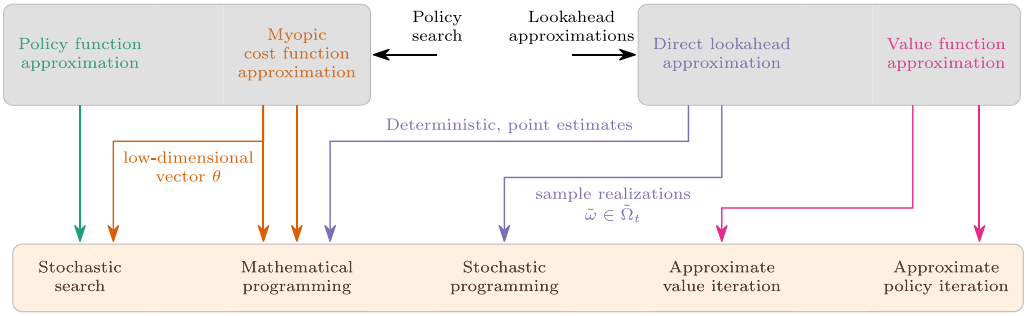

Fig. 3. Decision policies and algorithms. Decision policy categories are based on Powell [34].

steps for a fixed number of iterations N. In contrast, API performs multiple policy evaluations and uses this data to improve the policy. API involves two nested loops—an outer loop that controls the number of policy improvements M and an inner loop that controls the number of policy evaluations N. Both AVI and API simulate the decision making process within policy evaluation, removing the need to loop over all possible states.

AVI and API algorithms are widely used within ADP, and generally follow the aforementioned outlines; however, implementations tend to be problem specific—see Powell [3, Ch. 10] for several examples and Bertsekas [37] for an overview of API. Factors that affect an implementation include whether the problem is modelled as having a finite or infinite horizon, the type of value function approximation chosen, and how the approximation's parameter vector θ is learnt. Regarding the latter, many options exist that produce a good value function approximation (see Geist and Pietquin [38] for an overview), such as Least Squares Temporal Difference (LSTD) [39], Least Squares Policy Evaluation (LSPE) [40], or Support Vector Regression (SVR) [41]. In addition, within the mth policy improvement step an update equation of the form

$$\bar{\theta}^m = (1 - \alpha_{m-1})\bar{\theta}^{m-1} + \alpha_{m-1}\hat{\theta}^m, \tag{9}$$

is often used to update the estimate $\bar{\theta}^m$ for the parameter vector, where $\hat{\theta}^m$ is the most recent observation and α_{m-1} is a step size. There are many step size options, including constant, generalized harmonic, bias-adjusted Kalman filter, etc. [42]. As with the remainder of the algorithmic options, "strategies for stepsizes are problem dependent" [3, p. 451].

2.3.4. Evaluating ADP-generated policies

When evaluating an ADP-generated policy, a benchmark should be prepared because the algorithm selected to search for the policy or the policy itself may be poorly designed. Suggested benchmarks include: an optimal policy via DP; an optimal deterministic version of the problem; or a myopic policy [30]. In particular, the first option enables an ADP-generated policy to be compared to one that is optimal, however this is only feasible for relatively simple problems. Comparing to a myopic policy establishes the benefit of incorporating the downstream value in a decision policy. In addition, "direct comparison of [an ADP] algorithm's performance to earlier heuristic solutions should be made" [43, p. 10], where earlier solutions may include currently practices policies or previously published results.

Regardless of the benchmark selected, these comparisons may be done by simulating the implementation of the policies and comparing their objective function values, given in Eq. (1), and "[computing] the variance of the estimate of the difference to see if it is statistically significant" [3, p. 154].

3. Review methodology

To identify relevant articles for this review, Web of Science was searched on 9 April 2021 for articles published from 1995 onwards. Title, abstract, and keywords were searched using the following pattern:

- `(military OR army OR navy OR ''air force'') AND (''approximate dynamic programming'' OR ''adaptive dynamic programming'' OR ''neuro-dynamic programming'')`.

This search returned 16 results. We manually refined these results to those that focus on *military operations research*—"[t]he application of quantitative analytic techniques to inform military [or civilian] decision making" [27]—and were deemed to be an application-based article. This resulted in a set of 10 articles.

In addition, although the journals *Military Operations Research* and *The Journal of Defence Modelling and Simulation* are included in the Web of Science databases, the above search pattern did not return results from these journals. This is most likely due to the search pattern requiring `(military OR army OR navy OR ''air force'')`; thus, title, abstract, and keywords in each of these journals were searched using this pattern. Two articles from each were returned. One article from *Military Operations Research*, Flint et al. [44], focused on best practices for random number generation and variance reduction within the application of ADP algorithms to defence problems. Hence, it was deemed not to be an application paper and thus not within the scope of this review. Lastly, articles were added via references contained within these 13 articles. In total, 18 application-based articles were identified as relevant and are discussed in the following section.

4. Applications of ADP within military operations research

In this section, we present a summary of the 18 application-based articles identified through the aforementioned literature search. Table 2 lists each study reviewed, its application area, and characteristics of the ADP policies and algorithms implemented. The characteristics listed focus on those discussed in Section 2.3, namely:

- type of decision policy—Myopic CFA, PFA, VFA, DLA, or hybrid;
- value function approximation strategy—lookup table, parametric, or nonparametric;
- value function model—hierarchical aggregation, linear architecture, NN, etc.;
- algorithmic strategy—stochastic search, mathematical programming, stochastic programming, AVI, API;
- approach to updating the value function model parameters θ—temporal difference learning, LSTD, LSPE, SVR, etc.; and
- step size α_m—constant, generalized harmonic, polynomial, etc.

For some articles listed, not enough information was provided in order to identify how certain characteristics were addressed by the authors. In this case, the characteristic is listed as *Not specified*. In addition, for some articles listed certain characteristics are not applicable. In this case, the characteristic is listed as *N/A*. Further details are given below. Studies are organized into three categories—force development, force generation, force employment—and then listed in chronological order.

4.1. Force development

Within the force development category, two articles have been published: one that focuses on investment planning [45] and one on force structure analysis [46]. These two articles employ significantly different ADP strategies to solving their respective sequential decision problems, where the difference is rooted in the choice of decision policy.

Fisher et al. [45]: Every fiscal year, the Royal Canadian Navy selects a subset of candidate non-strategic capital projects that will receive funding, where each requires a fixed expenditure over a multi-year period and has an associated value to the Navy. Since each fiscal year's funding does not carry over to the next, historically Navy planners have tried to allocate the entire available budget while striving for high long-term value. Fisher et al. [45], with the aim to maximize the long-term value, designed a knapsack-based myopic CFA parameterized by a single scalar parameter α that reserves a portion of the budget, between zero and 100%, for future projects. The authors employed a simple enumeration method to assess the impact of the policy's parameter. The remaining characteristics listed in Table 2 are not relevant to this article, and thus listed as N/A.

The results obtained from this policy were compared to that obtained from a single optimization model that considered projects across the entire planning period simultaneously, i.e., the model knows how the future will unfold. Fisher et al. showed that for a 25 year planning period when $\alpha = 0.4$ (40% of the budget reserved) the myopic CFA was able to achieve 90% of the value found by the single optimization model, which was considerably better than the 73% obtained when α was set was zero (no budget reserved). The authors concluded that "planners who try to expend all of their planning space in the current year end up implementing a set of projects that have considerable lower value and higher cost" [45, pp. 88–89].

Southerland and Loerch [46]: The authors proposed an ADP algorithm based on AVI to support deliberations within United States Department of the Army's annual force structure review. This review, known as Total Army Analysis, aims to "identify changes, within total personnel constraints mandated by Congress, to the existing force structure that maintain or improve the Army's ability to meet Combatant Command mission requirements while maintaining the ability to respond to future crises" [46, p. 25]. The algorithm's objective included two weighted contributions: first, the force structure's ability to meet mission requirements over the following year; and second, the ability to meet readiness requirements. The authors employed a VFA based on diffusion wavelet transform [61], where the advantage of using this approach within an ADP algorithm is that "it generates the best basis functions as part of the approximation process [whereas] regression methods require the basis functions to be pre-specified" [62, p. 59]. In addition, the approach to learning the value function model's parameters is based on a three phase approach: *initialization* in which states are selected based on random decisions; *transition* in which a transition occurs from random decisions to VFA-based decisions; and an *exploitation* phase where states are selected based solely on the VFA. It should be noted that the ADP algorithm's pseudocode is not listed in this article; see Southerland [63, pp. 52–54] and Balakrishna [62, pp. 57–58] for details.

The authors conducted a set of experiments, varying the weights within the objective function and the number of sampled states used within the VFA to model the entire state space. The resulting ADP-generated policies were then compared to a myopic policy. To do so, 1000 20-year scenarios of mission and readiness requirements for 20 units types were generated. Each policy was then applied within each scenario three times, each time with a different initial force structure. With regards to meeting mission requirements, the ADP-generated policies showed an average improvement between 1.8% and 15.7% as compared to the myopic policy, whereas the average improvement to meet readiness requirements ranged between 7.6% and 19.9%. It is worth noting that increasing the number of states used in the diffusion wavelet transform approximation did not result in a better ADP-generated policy.

4.2. Force generation

One application within the force generation domain has been published [47]. This study, which focused on personnel sustainment, differs in its ADP strategy from those used to date in the force development domain; in particular, the use of a VFA that employs a parametric strategy and as a result the selected approach to update the value function model parameters. However, this approach aligns with those used in the force employment domain.

Hoecherl et al. [47]: In this study the authors investigated the problem of personnel recruitment and promotion decisions in the United States Air Force. The authors formulated two ADP algorithms to obtain decision policies. The first algorithm uses an API strategy with a VFA based on a set of unspecified basis functions in a linear

Table 2
ADP applications within military operations research during the period 1995–2021. Articles are divided by horizontal lines into three groups: force development (top group), force generation (middle group), and force employment (bottom group).

Article	Application area	Policy type	Approx. Strategy	Value function model	Algorithm strategy	Model update algorithm	Step size
Fisher et al. [45]	Investment planning	CFA	Parametric	N/A	Enumeration, mathematical programming	N/A	N/A
Southerland and Loerch [46]	Force structure analysis	VFA	Non-parametric	Diffusion Wavelet Transform	AVI	Explo-ration/Exploration	Not specified
Hoecherl et al. [47]	Personnel sustainment	VFA	Parametric	Linear/Separable piecewise linear	API/AVI	LSTD/Concave Adaptive Value Estimation (CAVE)	Decreasing
Ross et al. [48]	Situational awareness	VFA	Lookup table	Q-factor	AVI	Q-learning	Not specified
Bertsekas et al. [49]	Missile defence	VFA	Non-parametric /Parametric	NN/Linear	API	Least squares/TD(λ)	Decreasing
Popken and Cox [50]	Air combat	DLA	N/A	N/A	Game theory, Multistage stochastic program	N/A	N/A
Sztykgold et al. [51]	Battlefield strategy	VFA	Lookup table	N/A	AVI	TD(λ)	Not specified
Wu et al. [52]	Airlift	VFA	Parametric	Linear	AVI	N/A	Not specified
Powell et al. [18]	Airlift	VFA	Parametric	Piecewise linear concave	AVI	Not specified	Adaptive
Ahner and Parson [53]	Weapon target assignment	VFA	Lookup table	N/A	AVI	N/A	Not specified
Rettke et al. [54]	Combat medical evacuation	VFA	Parametric	Linear	API	LSTD	Harmonic
Davis et al. [55]	Missile defence	VFA	Parametric	Linear	API	LSTD	Harmonic
Laan et al. [56]	Illegal fishing patrols	VFA	Lookup table	Hierarchical aggregation	AVI	N/A	Harmonic
McKenna et al. [32]	Inventory routing	VFA	Parametric	Linear	API	Least squares/LSTD	Harmonic
Robbins et al. [57]	Combat medical evacuation	VFA	Lookup table	Hierarchical aggregation	API	N/A	Harmonic
Summers et al. [58]	Missile defence	VFA	Parametric	Linear	API	LSPE/LSTD	Harmonic
Jenkins et al. [59]	Combat medical evacuation	VFA	Parametric	Linear	API	SVR	Polynomial
Jenkins et al. [60]	Combat medical evacuation	VFA	Parametric/Non-parametric	Linear/NN	API	LSTD/NN	Polynomial

architecture, LSTD, and a decreasing step size to update the VFA's tunable parameters. The authors developed variants of this algorithm, specifically using instrumental variables in the regression equation as they "have been found to significantly improve regression performance" [47, p. 3080] and Latin Hypercube Sampling "to generate an improved set of post-decision states [and] help ensure uniform sampling across all possible dimensions" [47, p. 3080]. The second algorithm uses an AVI strategy, a VFA based on separable, piecewise linear value function approximations, and the CAVE algorithm [64] to update the approximations.

The authors compared the ADP-generated policies against the currently employed sustainment line policy in two 50-year scenarios of sustaining personnel: a small problem instance and a larger problem instance, where the latter was of particular interest to the Air Force. The difference between the two scenarios is defined in terms of number of career fields, number of officer grades, number of commissioned years of service, and number of new recruits. Across the two problem instances the effect of changing several scenario and algorithmic parameters were evaluated, including order of the basis function, number of hiring and promotion decisions, and inclusion or exclusion of the aforementioned variants. Overall, the LSTD-generated policy performed best in the smaller problem instance, showing approximately an 8% improvement in the objective function (reduction in cost) as compared to the benchmark policy when using fourth-order basis functions and the two previously mentioned variants. However, this algorithm performed significantly worse than the benchmark policy in the larger problem. The CAVE-generated policy performed better in the larger scenario instance, showing an improvement of 2.8%.

Although not published in a peer-reviewed journal or conference, it should be noted that similar problems have been studied by Bradshaw [65] and West [66] as part of their graduate programs at the Air Force Institute for Technology, and Situ [67] at George Mason University.

4.3. Force employment

Within the force employment category, 15 articles have been published. Combat medical evacuation is the topic with the most publications [54,57,59,60], with the remaining articles covering a wide variety of topics including airlift [18,52], missile defence [49,55], and inventory routing [32]. All studies in this category developed policies based on VFA, with the exception of Popken and Cox [50] which used a DLA. Within those studies that used a VFA, all three approximation strategies—lookup table, parametric, non-parametric—were used, and six used an algorithm based on AVI with the remaining eight being based on API.

Ross et al. [48]: The requirements of a Commander's situational awareness evolve over time, and thus information fusion strategies must adapt to new information and new requirements. In this application, Bayesian networks were used to infer, based on a variety of data sources, the type of military unit formed by clusters of military vehicles. Under the assumption that available computational resources are insufficient to evaluate all incoming data in the Bayesian network, the ADP algorithm's aim is to optimize the use of the data to maximize the knowledge about the battlespace in order to inform a military commander. The ADP algorithm is based on Q-learning [13, pp. 131–132], however a detailed algorithm is not presented within this article.

The algorithm was tested in a scenario with up to five military unit types operating within a 75 km by 90 km area. Simulated training data was used to train the ADP algorithm, where on average at any time there were 14 clusters of military vehicles present and seven false alarm clusters. The results indicated the ADP algorithm outperformed a random controller, increasing the probability of correct classification to 0.53 from 0.31.

Bertsekas et al. [49]: This study examined a theatre missile defence problem in which the attacker has a limited inventory of multiple types of missiles, the defender has multiple types of interceptors where the number launched at a given time is limited by the of number launchers, and the defender must allocate specific interceptors to specific missiles. With the aim to maximize the expected value of targets surviving at the end of the battle, the authors describe an API algorithm that used a non-parametric VFA based on a NN with a single hidden layer, as well as an API algorithm that used a VFA with a linear architecture. Several variations to compute the estimate of the weights within both were explored: Monte Carlo simulation and least squares; temporal difference learning (TD(λ)); and one labelled as *optimistic policy iteration* which is similar to an AVI algorithm. A decreasing step size is then used to update the weights.

The authors compared their ADP-generated policies within a representative scenario, which included one attack missile type, one interceptor missile type, and three targets to be defended. In this scenario, the authors used four features to approximate a state's value: missile leakage, surviving target value, number of targets, and number of interceptors. Twenty-three test cases were developed that spanned a wide range of scenario instantiations, including those that overwhelmed the defender or the attacker, each with a different degree of interceptor effectiveness. For each test case, results for the API algorithm using Monte Carlo simulation and a NN, Monte Carlo simulation and a linear architecture, and the optimistic policy iteration algorithm were reported. As each algorithm includes a variety of parameter settings, the authors used "a mixture of insight and preliminary experimentation to arrive at a combination of settings for each algorithm that would work robustly" [49, p. 47]. The articles lists some, but not all, of the parameter settings. Results were compared to both an exact decision policy and a heuristic. Although a detailed comparison of the test cases was not reported, amongst the authors' conclusions it is worth noting that "algorithm performance is highly dependent on the tuning of these parameters" [49, p. 50] and that no single VFA was superior across the tests conducted.

Popken and Cox [50]: In this article the authors examined the modelling of air combat to support air warfare operational planning in an effort to demonstrate "embedding of optimization algorithms [that account for the complexities and uncertainties in real-world conflicts] into an operational planning cycle operating over a multi-period conflict can be made practical" [50, p. 127]. The authors modelled the problem as a stochastic two-player game, implemented as a simulation–optimization, with a hierarchy of decisions, i.e., force allocation decisions down to flight control decisions. Within the force allocation level, at each decision epoch aircraft were assigned to one of five roles: counter air, air defence, target reduction, anti-aircraft fire suppression, or other. These decisions are generated by a DLA via a multistage stochastic program. Target assignments were then based on a greedy heuristic and a stochastic simulation was used to determine the outcomes.

In testing, the authors utilized a representational scenario of an air conflict between the United States and North Korea; in particular, 15 base locations for United States forces positioned in and around South Korea, and 201 targets consisting of various infrastructure types in North Korea. ADP-generated policies were generated through varying two algorithmic parameters and one scenario parameter. The two algorithmic parameters were the number of improvement steps in the simulation–optimization algorithm, and horizon weight which balanced the need for forces to survive in the near term versus the long term. The one scenario parameter varied was the quality of the intelligence assessment, which influenced the accuracy of information and thus decisions on how to allocate aircraft. The authors concluded that the optimal tradeoff between time and performance occurred around 100 improvement iterations, a lower horizon weight to not overshoot enemy targets, and poor intelligence quality which contributes to lower horizon weight and balances value against survivability.

Sztykgold et al. [51]: This study explored decision-making in a two-sided terrestrial conflict where the goal is to determine a battle strategy for one side to control a target location while maintaining a minimum strength ratio between friendly and adversarial forces. The authors modelled the problem as a stochastic game in a multi-layered graph, and designed an algorithm based on temporal difference learning (TD(λ)) which employs a variation of generalized policy iteration [13]. For the purposes of this review, we have classified this algorithm as an AVI using VFA based on a lookup table, and the value function model as N/A since the authors do not describe any form of state aggregation. The algorithm's pseudocode is not provided in the article.

In testing, the AVI algorithm's result was compared with two optimal policies found via enumeration. This was possible due to the small size of the scenario, which included 16 vertices and a strength ratio constraint of two to one for allies to enemies. The authors ran three experiments, each with 25 trails with 1000 iterations of the algorithm and varied λ to affect how learning was performed. The results showed that "more than 84% of trials return the optimal control, and the value of λ does not change this ratio" [51, p .6]. The authors did not provide further details regarding the experiments.

Wu et al. [52]: In this article the authors discussed a variety of existing approaches to optimize military airlift, and presented an ADP approach to improve solution quality for the United States Air Mobility Command. With the aim to maximize cargo throughput, the authors describe an ADP policy that uses a linear approximation centred around the post-decision state variable. In addition, the authors describe how the policy may be extended to incorporate expert knowledge. Of note, while the authors describe how the value estimate of the post-decision state variable may be updated via Eq. (9), they neither specify the step size nor provide the pseudocode for the ADP algorithm implemented. However, given the description in the article it is plausible that the algorithm used is similar to the *Value iteration using a post-decision state variable* described by Powell [3, p. 391]. For this reason, in Table 2 we have chosen to list the algorithm as AVI and the model update algorithm as N/A due to that a specific algorithm, other than Eq. (9), is used to update the value estimate of the post-decision state variable.

In testing, the authors utilized a representative scenario of managing six aircraft types to move passengers and cargo between the United States and Saudi Arabia, where the total requirements were four times the total capacity of all the aircraft. To integrate the risk of breakdowns, the scenario assumed a failure probability of 20% on the C-141B freighter with an associated five day repair period, and that all other aircraft could be repaired without delay. The policy generated via the API algorithm was compared to four other policies, which in order of the amount of information considered are:

- a rule-based policy that examined one requirement and one aircraft at a time;
- a myopic cost-based policy that looked at one requirement and a list of aircraft, knowable now and actionable now;
- a myopic cost-based policy that looked at a list of requirements and a list of aircraft, knowable now and actionable now; and
- a myopic cost-based policy that looked at a list of requirements and a list of aircraft, knowable now and actionable in the future.

The results showed a maximum improvement in the objective function of 100% for the policy generated via the API algorithm as compared to the first policy (rule-based policy) and a minimum improvement of 10% compared to the fourth policy. The policy generated via

the API algorithm also outperformed the four policies by reaching the maximum throughput in a fewer number of time periods, around 80 time periods as compared to a maximum of 270 for the rule-based policy. Overall, the study concluded that by increasing the informational content in a policy, the solution quality increased in terms of both maximized throughput and minimized cost.

Powell et al. [18]: The authors developed a dynamic program that describes the assignment of military airlift to pickup and move passengers and goods, or move empty to another location, to meet prescheduled requirements or those that dynamically arise over time. The authors designed an AVI algorithm, whose objective is to maximize the reward received for fulfilling airlift requirements, that used a VFA based on a piecewise linear and concave function centred around the post-decision state variable. This approximation strategy was selected as it has been shown to work well in fleet management problems—see Powell [17],Wu et al. [52],Godfrey and Powell [64]. The value estimate of the post-decision state variable is updated via an equation similar to Eq. (9), where an adaptive step size is used [42]. Following this update, corrections are made for concavity violations using one of many available approaches [68], although the one used in this study is not identified. For this reason, we have listed the model update algorithm as Not specified.

The authors perform a series of experiments within the context of providing military airlift within Canada; specifically, a fleet of 13 aircraft servicing demands that may arise across 401 locations during a one month period, and decisions regarding which airlift will fulfil which requirement being made every six hours. Experiments included both deterministic and stochastic cases. For deterministic demands, the experiments showed almost no benefit to using future value when making decisions since most requirements could be fulfilled within six hours, and essentially all within 12 h. When stochastic demands are included, which arose with 0, 2, or 6 hours notice to move, the experiments showed that the ADP-generated policy outperformed a myopic policy by approximately 5% when stochastic demands needed to be immediately moved; otherwise, little difference was observed. Lastly, experiments that included both random aircraft failures and stochastic demands showed that the ADP-generated policy outperformed a myopic policy in all combinations considered.

Ahner and Parson [53]: This article studied the sequential allocation of different weapons to adversarial targets. Although engagements may occur within a variety of contexts, in this study the platforms carrying the weapon systems proceeds through a series of stages, where at each stage a platform may engage a variety of targets and may be destroyed by the adversary. The authors presented an AVI-based algorithm and employed a VFA based on the post-decision state variable. For the purposes of this review, we have classified the value function model as N/A since the authors do not describe any form of state aggregation. In addition, we list the model update algorithm as Not specified—while the authors indicate that the value function is updated using Monte Carlo samples, the precise approach is not listed.

The authors discussed two applications. First, a simple case with two weapon systems and two targets; and second, a larger combat simulation, although specific details are not provided. The simple case, in which three scenarios are analysed, is solved via value iteration due to its size. Detailed results for the larger case are not presented.

Rettke et al. [54]: In this article the authors studied the dispatch of Medical Evacuation (MEDEVAC) units to pickup injured personnel in the battlefield and transport them to a medical treatment facility. With the aim to maximize the value of calls covered, the authors developed an API algorithm that employs a VFA based on a linear architecture with six features—unit availability, expected time until unit availability, expected time of arrival to the nearest medical treatment facility, expected total time of a call in the system if served, priority of a call, and expected call time in system if assigned. These features are combined to create polynomial functions, up to third order, to approximate the post-decision state value function. Using data collected via a number of policy evaluation iterations, the algorithm computes $\hat{\theta}$ using LSTD and updates θ using a generalized harmonic step size.

The authors tested their strategy in a representative scenario in which a coalition of allied countries are performing a peacekeeping mission in northern Syria. In particular, the scenario includes 18 locations where casualties are expected to occur, and five locations were MEDEVAC units are stationed, of which two have medical treatment facilities co-located. In testing, the ADP-generated policy was compared to the currently practiced policy, which dispatches the nearest MEDEVAC unit regardless of future considerations. This comparison was done across 27 scenario instances, where one scenario parameter was varied (the arrival rate of casualties) and two algorithmic parameters were varied (number of policy improvement iterations, number of policy evaluation iterations). The ADP-generated policy showed a 31% improvement when using a third-order polynomial VFA as compared to the currently practiced policy in the baseline scenario, i.e., one casualty per hour, 10 policy improvement iterations, 20 000 policy evaluation iterations. It is worth nothing that as the casualty frequency decreased, the ADP-generated policy's gains over the currently practiced policy became negligible.

Davis et al. [55]: This study examined the fire control policy of the United States military's missile defense interceptors; specifically, the authors supported decision makers through developing policies to determine how many interceptors to launch at each incoming missile. With the goal of maximizing remaining city value after a conflict, the authors developed an API algorithm that employed a VFA based on a linear architecture with seven features, including duration of conflict, the battlefield damage assessment capabilities of the attacker, and the interceptor inventory of Surface to Air Missile (SAM) silos. Within the algorithm, LSTD was used to compute the estimate $\hat{\theta}$ and a generalized harmonic step size is used to update the estimate for θ.

The authors conducted experiments set within the context of a representative scenario with three homogeneous cities being defended by two SAM batteries. Specifically, the defense salvos were positioned to provide overlapping protection for the middle city, have an inventory of ten interceptors, and have a firing limit of four interceptors at a time. In testing, the authors compared the ADP-generated policy against one generated using Gauss–Seidel modified policy iteration. These policies were compared in four test scenarios, with 2430 instances in each being executed. Two scenario parameters were varied (expected number of attacks/conflict duration, battlefield damage assessment capabilities of attacker) and four algorithmic parameters were varied (number of policy iterations, number of policy evaluations, number of chosen basis functions, and a single scaler parameter within the step size function). The ADP-generated policy performed with a mean optimality gap in the range of 8%–22% over the four test scenarios, depending on the tuning of the conflict duration and inventory sizes. The optimality gap was lower for conflicts with shorter duration, those with smaller defender SAM inventories, and with fewer surviving cities.

Laan et al. [56]: The authors studied a security game whose aim is to determine a patrolling strategy to protect a large geographical area against an intruder, under the constraint that specific subregions must be patrolled multiple times. The authors modelled the problem as a stochastic game and designed an AVI algorithm that employed a VFA, based on hierarchical aggregation, around the post-decision state variable. The value estimates were updated using a harmonic step size. In addition, the algorithm used a ϵ-greedy exploration–exploitation approach to encourage exploration of states not yet visited. The model update function is listed in Table 2 as N/A since an approach beyond Eq. (9) is not used.

The ADP-generated policy was compared with a static policy, i.e., one that is fixed at the beginning of the game, and a dynamic policy within an illegal or unreported fishing scenario. Two versions of the scenario were analysed: a small and a larger instance, where the small instance allowed policies to be solved to optimality. The authors explored the effects of algorithmic parameters, including initial step

size, the balance between exploration and exploitation, and the number of aggregation levels. For the small instance, the results demonstrated the ADP-generated policy outperformed the static policy, but worse than the dynamic policy. However, the authors noted this was not always the case; in particular, in scenario instances where the values of static and dynamic policies were close. The results for the large instance are presented, but are not compared to a benchmark.

McKenna et al. [32]: In this article the authors developed an API algorithm to search for policies to improve the resupply of geographically dispersed combat outposts within an austere environment via cargo unmanned aerial vehicles. The authors employed a VFA that, based on the post-decision state variable, used a linear architecture with four features—the inventory at each outpost; number of operational unmanned aerial vehicles; threat condition level, where a high-risk level corresponds to a higher probability of a failed delivery; and the number of unmanned aerial vehicles deployed to each outpost. Two approaches were explored, least squares and LSTD, to compute the estimate $\hat{\theta}$, and a generalized harmonic step size rule was employed.

The authors evaluated their approach within a representative scenario. Specifically, unmanned aerial vehicles from a single brigade support battalion were used to resupply outposts for up to a three month period, where each outpost represented approximately 50 personnel requiring 2000 pounds of supplies per six hour time period. Four problem characteristics and four algorithmic features were investigated:

- **Problem characteristics**: number of outposts; initial number of unmanned aerial vehicles; probability of remaining in a low threat environment; probability of remaining in a high threat environment; and
- **Algorithmic features**: number of policy evaluation loops; number of policy improvement loops; least squares or LSTD; and whether smoothing is used to update θ or not.

Based on the 68 experimental runs, the results indicated that in 31% of the runs the ADP-generated policy significantly outperformed the myopic benchmark policy. In addition, for those runs that used LSTD and smoothing this percentage increased to 76%. Regarding problem characteristics, the results showed that the ADP-generated policy's performance decreased as the number of outposts increased, but was robust in terms of other characteristics. As an example, results indicated that the ADP-generated policy was able to supply 57% of outpost demand over a three month period as compared to 18% using a myopic policy.

Although not published in a journal or conference, this topic was also studied by Salgado [69] as part of their graduate program at the Air Force Institute of Technology.

Robbins et al. [57]: The authors studied the United States Army's aeromedical evacuation system, specifically the sequential resource allocation decision problem of dispatching evacuation assets to respond to casualty events and transport individuals to mobile hospitals. The authors proposed an API algorithm that employed a VFA, based on hierarchical aggregation with three levels, around the post-decision state variable. This method was selected as the United States Army uses a zone scheme to characterize requests, and "any proposed real-world modifications to policy are more likely to be heeded if they adhere to contemporary procedures and frameworks currently in use" [57, p. 3]. The value estimates at each aggregation level are updated using a generalized harmonic step size. In addition, the algorithm uses a ϵ-greedy exploration–exploitation approach. Similar to Laan et al. [56], we label the model update function as N/A in Table 2.

The authors compared their ADP-generated policy to two benchmarks: first, the optimal dispatch policy (via dynamic programming within two small-scale problem instances); and second, the typically used closest-available dispatch policy. The comparison was done in the context of the United States military performing combat operations in Afghanistan; specifically, in a region containing two main operating bases, two forward operating bases, and 56 casualty clusters. Three zone configurations—6, 12, and 34—were tested, and for each configuration six scenarios were generated with a different system state. For the 12 zone configuration, the ADP-generated policy attained at least a 1% optimality gap, representing approximately a 9% improvement over the typically used policy. For the 34 zone configuration, the ADP policy attained a 12.4 ± 0.4 percent improvement over the same benchmark policy.

Summers et al. [58]: In this article the authors developed two API algorithms to search for firing policies in a networked missile defense system against theatre ballistic missiles for the United States Air Force. In particular, the authors consider a combination of three systems—the Terminal High Altitude Air Defense, the Aegis Ballistic Missile Defense System, and the Patriot air defense system. With the aim of minimizing protected asset damage and expected total cost, the algorithms employed a VFA that utilized six basis functions in a linear architecture. The first algorithm used LSPE to update the estimate $\hat{\theta}$, while the second used LSTD. Both algorithms used a generalized harmonic step size.

The two algorithms were tested in a representative scenario: two assets are attacked by a combination of theatre ballistic missiles and multiple re-entry vehicles with the Aegis system intercepting the mid-course phase, and Terminal High Altitude Air Defence/Patriot systems intercepting the terminal phase according to their co-located asset. A duo of myopic firing policies, one being the currently applied United States doctrine, are compared with the ADP-generated policies across 86 test instances. First is the Match policy, which fires one interceptor per attacker missile, and second is the Overmatch policy used by the US military which fires two interceptors if possible. Testing examined the effects of varying both scenario parameters and algorithmic parameters:

- **Scenario parameters**: duration of the attack; enemy level of weapon sophistication; defender level of technological development; defended asset value; and
- **Algorithmic parameters**: number of policy iterations, number of policy evaluations, number of selected basis functions, a scalar parameter within the harmonic step size function, and a regularization parameter.

Overall, both the LSPE and LSTD algorithms outperformed, up to 25%, the baseline policies when the conflict duration was short and the enemy had more developed weapons. In scenarios with a long conflict duration and in which the defender had higher quality weapons, the Match policy performed best. Notably, the Overmatch policy, which is the current United States implemented doctrine, was never the best policy choice.

Jenkins et al. [59]: This article examined how the United States Army conducts aeromedical evacuation during combat operations, and designed an API algorithm to determine a policy for the dispatching, preemption-rerouting, and redeployment of aeromedical helicopters. With the aim to reduce the time to transfer a casualty from the battlefield to a medical treatment facility, the strategy uses a VFA with a linear architecture and six features: the status of each evacuation unit; the remaining service time associated with each evacuation unit-service request pair; precedence level associated with each evacuation unit-service request pair; precedence level associated with each request not currently serviced; the total time each request has existed; and the total distance required for each unit to service each request. SVR is used to compute the estimate $\hat{\theta}$, as it "seeks to directly maximize generalizability", "is modelled as a convex optimization problem", "is relatively robust to outliers", and "has relatively few critical parameters that need to be tuned to obtain state-of-the-art results" [59, p. 137]. Lastly, the algorithm uses a polynomial step size.

The authors compared the ADP-generated policies with two benchmark policies: first, the currently used closet-available dispatch policy; and second, an extension of the current policy that allows preemption

and rerouting that we label as myopic. The comparison was done within the context of a fictional scenario in which the United States military is performing combat operations in support of the Azerbaijani government. The scenario includes two main operating bases (with a medical treatment facility and staging facility), two forward operating bases (staging facility only), and 55 centres where casualties are likely to occur. ADP policies were generated for a variety of algorithmic settings, e.g., number of policy evaluation loops, resulting in up to a 9.6 ± 0.2 % increase over the first benchmark and up to a 6.6 ± 0.2 % increase over the second benchmark. In addition, the authors demonstrated that as the rate at which casualties requesting evacuation decreased, so did the improvement of the ADP-generated policies over the benchmark policies indicating no significant benefit during low intensity conflicts.

Jenkins et al. [60]: The authors examined how the United States Army conducts combat medical evacuation through dedicated medical platforms with onboard medical professionals. The authors developed two algorithms based on API, each with the objective to find a policy to dispatch and redeploy helicopters which minimizes the time required to transfer casualties from the battlefield to medical treatment facilities. Each algorithm employed a VFA with eight basis functions, including: the availability status of each MEDEVAC unit; the expected time from the current time until each MEDEVAC unit transfers its onboard casualty to the nearest medical facility; etc. The first algorithm combined the eight basis functions using a linear architecture, LSTD to compute the estimate $\hat{\theta}$, and a polynomial step size. The second algorithm used a three-layer feed-forward NN (with one hidden layer), using scaled basis function evaluations as inputs, to estimate the value function. A back-propagation approach is used to compute a sample estimate of the network's parameters, and a polynomial step size is used to update the parameters.

The authors compared the ADP-generated policies with the currently practiced closest-available dispatch policy. This was done within the context of a representative scenario, in which the United States military is performing high-intensity combat operations in support of the government of Azerbaijan. The scenario included two main operating bases, which both have medical treatment facilities, two forward operating bases, and 55 casualty clusters. The authors examined 30 problem instances, varying specific scenario and algorithmic parameters throughout: casualty arrival rate; proportion of urgent, priority, and routine requests; number of policy evaluation iterations, etc. ADP-generated policies using the LSTD and NN approach significantly outperformed—up to 346% in the case of the latter—the closest-available dispatch policy in 24 and 27 of the 30 problem instances respectively, with gains increasing as the average request arrival rate increased.

5. Discussion

Based on the summaries in the previous section, this section presents five topics—classes of problem studied, evaluating ADP-generated policies, the value of incremental policies, policy robustness, and additional application areas—and provides recommendations within the last four topics.

5.1. Classification of problems studied

Two broad types of problems may be solved by ADP-based strategies: *state independent* and *state dependent*. State independent problems are those in which "the state variable captures only our belief about an unknown function, but where the problem does not depend on the state variable", whereas state dependent problems are those where the "contributions, constraints, and/or transition function depend on dynamically varying information" [34, p. 797]. In addition, Powell [34] has also proposed that each class may be further divided based on whether their objective function focuses on the final contribution or the cumulative contribution. The result is four classes of problems.

Table 3
Summary of objective function type in problems surveyed. Articles are divided by horizontal lines into three groups: force development (top), force generation (middle), and force employment (bottom).

Type of objective function		Article
Final	Cumulative	
	•	Fisher et al. [45]
	•	Southerland and Loerch [46]
	•	Hoecherl et al. [47]
•		Ross et al. [48]
•		Bertsekas et al. [49]
	•	Popken and Cox [50]
•		Sztykgold et al. [51]
	•	Wu et al. [52]
	•	Powell et al. [18]
	•	Ahner and Parson [53]
	•	Rettke et al. [54]
	•	Davis et al. [55]
	•	Laan et al. [56]
	•	McKenna et al. [32]
	•	Robbins et al. [57]
	•	Summers et al. [58]
	•	Jenkins et al. [59]
	•	Jenkins et al. [60]

All 18 problems reviewed are classified as state dependent. Within this set, 15 used an objective function focused on a cumulative contribution and the remaining three used a final contribution as listed in Table 3. Those articles that focus on a cumulative contribution span the force development, force generation, and force employment domains. For example, Fisher et al. [45] seeks to maximize long-term investment value, Hoecherl et al. [47, p. 3075] seeks to "improve policies regarding management of the commissioned officer corps", and Jenkins et al. [60] aims to maximize the response provided to battlefield casualties. In contrast, those articles that focus on a final contribution exist solely in the force employment domain, such as Sztykgold et al. [51] in which the objective was to position an army at a given location before a specified time.

Intuitively, this breakdown aligns with the type of activities that occur within these domains. Force development is concerned with implementing changes to existing capabilities, investing in new capabilities, and so on in order to provide and sustain effects for a designated period; force generation is concerned with organizing, training, etc. a force in order to maintain a readiness posture, sustain capabilities, and provide flexibility in case of contingency operations; and force employment is concerned with command, control, and sustainment of a commander's allocated forces [70]. Given these definitions, the objective of many force development and force generation activities are ongoing, and their objective is better described by a cumulative contribution rather than a final contribution. In contrast, the objective of a force employment activity may focus on either a cumulative contribution or a final contribution depending on the mission's objectives. For example, Bertsekas et al. [49] studied a missile defence scenario in which the objective is to maximize the value of assets surviving at the end of the scenario. Davis et al. [55] and Summers et al. [58] also studied a missile defence scenario; however, rather than their objective being to maximize the final value of the defender's assets, their objectives were designed to minimize the loss of assets' value over time such that assets that survive longer provide greater value.

However, this separation of objectives amongst force development, generation, and employment is not clear-cut. For example, while many defence investment planning problems focus on maximizing cumulative contribution over a long planning horizon [71–73], an alternative objective, such as minimizing a military's maximum capability gap after a specific time-frame, may be used. In this case, the objective is focused on the final rather than the cumulative contribution. Thus, while this review demonstrates a potential mapping between the three domains and the type of objective function, future studies may not follow this pattern.

5.2. Evaluating ADP-generated policies

Recall that suggestions exist on how to evaluate an ADP-generated policy, including the development of a benchmark policy, comparing the ADP-generated policy's objective function to the benchmark's value, and testing for statistical significance. Out of the 18 articles reviewed, seven followed these three suggestions as listed in Table 4. As an example, Robbins et al. [57] compared the objective function values of the reported ADP-generated policies to those attained by a currently practiced policy and showed a statistically significant 12.4 ± 0.4 percent improvement at the 95% confidence level in the 34-zone problem instance. In contrast, Fisher et al. [45] compared the objective function values of the reported ADP-generated policy with that of a single optimization and showed the former achieved at best 90% of the latter's objective, however the authors did not compute the variance or test for statistical significance. In addition, the type of policy selected as a benchmark varied throughout the reviewed articles, ranging from a single optimization model as in Fisher et al. [45] to using both a currently practiced policy and a myopic policy as in Summers et al. [58] and Jenkins et al. [59].

This review demonstrates a lack of standard practice to evaluating an ADP-generated policy, which makes it difficult to judge the quality of a solution. Within a defence context, there is significant advantage to using a currently practiced policy as a benchmark. Such a comparison demonstrates to a military's leadership the potential benefits of modifying existing doctrine, something which may meet resistance especially during peacetime [74]. Thus, it is suggested that for those studies in which ADP is being applied to a new topic or problem, a currently practiced policy be used as a benchmark if one exists. If one does not exist, then one or more policies, such as myopic, random, industry standard (e.g., in mass evacuation scenarios, the simple triage and rapid treatment (START) policy often used in mass-casualty scenes [75]), or other reasonable policies should be set as benchmarks. In addition, follow-on ADP-generated policies should be compared to relevant previously published studies in order to demonstrate the newly proposed policy's value.

5.3. Incremental versus comprehensive policy changes

There are a wide variety of policies that can be generated via ADP, and when compared to a current practiced policy run the gamut from an incremental change to a complete overhaul. These changes can be measured along two axes: data and algorithms. Fisher et al. [45], whose work focused on capital investment, is an example of an incremental change in terms of data requirements and a large change in terms of modelling. The proposed policy employed a single parameter not used in the current policy, and used a knapsack-based formulation as opposed to the current manual project selection method. In contrast, Jenkins et al. [59], whose work focused on medical evacuation dispatch, is an example of a large change in terms of both data and algorithmic approach. The authors designed two policies, one NN-based policy and one LSTD-based policy, whose input data each consisted of eight basis functions as compared to the current closest-available unit dispatching policy that only considered a single input. Robbins et al. [57], which also focuses on medical evacuation dispatch, is another example of a large change in data requirements and algorithmic design. However, in this case the authors based their VFA, which used hierarchical aggregation of geospatial zones, on current United States military zoning practices for battlefield medical evacuation. As stated by Robbins et al. [57, p. 3], "keeping problem complexity [in terms of data] and proposed solution techniques somewhat simple at first, and incrementally improving the sophistication of the modelling and solution techniques over time enables researchers to more effectively interact with practitioner communities". Taken a step further, starting with policies that are incremental when compared to those currently practiced not only supports interaction but also may increase the likelihood of deploying the policies into the field.

The applications discussed in this review overwhelmingly focus on a single domain, and while at times do represent a significant change in data requirements when compared to currently practiced policies, the required data does not tend to span multiple organizations across a defence enterprise. When seeking to apply ADP to multi-domain sequential decision problems, such as multi-domain command and control, it is likely that the number of required data sources will further increase and in many instances these sources will be siloed. Data silos have a negative impact on the development and deployment of artificial intelligence [76], including ADP-generated policies. The negative impact includes not only analysts spending the majority of their time collecting and cleaning data rather than modelling and interpreting results [77], but also can lead to unintentional ignoring of valuable data [78]. The recent Joint All-Command and Control project within NORAD is an example of an approach to tackle data silos that exist within defence [79]. Even though breaking down data silos allows for access to richer sources of data in a timely fashion, ADP-generated policies that are based on incremental policy changes will still have value in terms of faster implementation due to the fact that designing policies is an art form and incremental changes provide a limited space to seek improvements [3].

5.4. Policy robustness in defence scenarios

In articles whose primary motivation is direct application for specific scenarios, there exists three types of parameters that can be varied in ADP: policy parameters (the tunable vector θ); algorithmic parameters (i.e., number of policy improvements, step size, discount rate, etc.); and scenario parameters (e.g., the arrival rate of casualties, adversary's technical sophistication, etc.). In a military context, prior to deploying an ADP-generated policy it is important to ensure the policy is *robust*—that is the policy performs or works well in a variety of plausible futures or scenarios [80]. While the algorithmic parameters are used to tune the policy parameters, it is also important to vary the scenario parameters when evaluating a policy. These evaluations vary across the papers reviewed. On one end of the spectrum, Fisher et al. [45], which focused on defence project selection and employed a myopic CFA, varied the policy's tunable parameter but did not consider changes to the algorithmic or scenario parameters. The authors opted to modify the amount of budget reserved for future years and each test instance employed the same fixed conditions such as the pattern of project arrival value. In contrast, Davis et al. [55], which focused on missile defence and employed a VFA, varied four algorithmic parameters. While the articles reviewed differ in their approach to evaluating a policy, those more recently published tend to incorporate multiple scenario instances in their evaluation.

In addition to robustness, a few articles provided insight into the impact of varying scenario parameters. Summers et al. [58], which focused on missile defence, varied several scenario parameters when evaluating their ADP-generated policies' performance. The authors probed into the effects of conflict duration, the level of weapon sophistication of both the attacker and defender, and the values of the assets defended. Their testing showed that when the conflict duration was short and the attacker weapons were sophisticated, the ADP-generated policies performed best with an improvement of up to 25% against the current policy. In contrast, when the conflict was long and the defender's weapons were more sophisticated, the ADP-generated policies only showed a gain of around 5%. Jenkins et al. [60], which focused on combat medical evacuation dispatch, explored the effects of a single scenario parameter, the arrival rate or conflict intensity, had on the performance of their ADP-generated policy. The results showed that when the arrival rate of casualties was one per hour, the ADP-generated policy gave almost no benefit over the myopic policy, but when the arrival rate was once every 20 min, there was an almost 18% improvement.

Table 4
Summary of reported approaches to evaluate ADP-generated policies. Articles are divided by horizontal lines into three groups: force development (top), force generation (middle), and force employment (bottom). Current indicates a currently practiced policy, Myopic indicates a myopic policy or an extension of a currently practiced policy that is myopic, Exact indicates a DP algorithm was used to find an optimal policy, and Other indicates a single optimization model, a rule-based policy, etc. Note: Rettke et al. [54], Robbins et al. [57], and Jenkins et al. [60] describe the benchmark policy used as both myopic and currently practiced. These are labelled as Current rather than Myopic.

Benchmark policy				Objective function	Statistical significance	Article
Current	Myopic	Exact	Other			
			•	•		Fisher et al. [45]
	•			•		Southerland and Loerch [46]
•				•	•	Hoecherl et al. [47]
			•	•		Ross et al. [48]
		•	•	•		Bertsekas et al. [49]
						Popken and Cox [50]
		•		•		Sztykgold et al. [51]
	•		•	•		Wu et al. [52]
	•		•	•		Powell et al. [18]
						Ahner and Parson [53]
•				•	•	Rettke et al. [54]
		•		•		Davis et al. [55]
		•	•	•		Laan et al. [56]
	•			•	•	McKenna et al. [32]
•		•		•	•	Robbins et al. [57]
•	•			•	•	Summers et al. [58]
•	•			•	•	Jenkins et al. [59]
•				•	•	Jenkins et al. [60]

This demonstrated that the value of planning for the future decreased as conflict intensity decreased and that a simple naive or myopic policy would perform just as well under those circumstances. These articles demonstrate that when scenario parameters are varied those scenario instances in which specific policies show benefit may be identified. Since real-world applications of ADP-generated policies cannot expect environmental conditions to remain constant, policy performance and selection is scenario-dependent. If ADP-generated policies are to inform doctrine and be deployed in a military context, the policies generated must be robust and thus it is recommended that policies be evaluated against a range of scenario parameters, or the scenarios themselves be non-stationary.

5.5. Additional topics within defence

The studies reviewed in this article span the spectrum of force development, generation, and employment. However, the emphasis to date has been on force employment applications, with 15 of the 18 articles fitting within this category, two fitting within the force development category, and only one within the force generation category. Given this distribution, a natural question is whether potential applications areas exist within the two underrepresented categories.

Within force development there are several decisions that occur within defence that are aptly described as sequences of decisions over time under uncertainty. Such decisions may benefit from applying ADP strategies to develop improved decision policies as compared to those currently practiced. Examples, along with similar applications outside defence or articles within defence that suggest ADP as an approach, include: repair and replacement of major platforms, including mid-year life upgrades, such as discussed in Stasko and Gao [81], Abdul-Malak and Kharoufeh [82], and Sadeghpour et al. [83]; supply chain management, including sourcing strategies [84], green logistics in remote regions such as the Arctic [85], and munitions planning [86]; and investment planning, such as community resilience planning as applied to military bases [87] and long-term defence-wide capital investment planning [72,88]. It should be noted that Fisher et al. [45] applied ADP within the defence investment planning, but the application was limited to a single service.

Within force generation opportunities exist to apply ADP methods as well. Examples include: recruiting and promotion of non-commissioned military members, thus extending the work by Hoecherl et al. [47]; selection of joint exercise and training events over a multi-year planning horizon in order to meet joint readiness objectives [89]; and military occupation sustainability, such as pilots [90]. Although the latter focuses on using simulation and does not specifically mention ADP, the author states the model has been used to determine “optimal fleet intake/absorption levels, how to optimally plan the transition of platforms or the introduction of a new capability” [90, p. 53] and notes that several exogenous factors affect pilot attrition, such as pilot recruiting programs by industry, and changes in aircraft yearly flying rate. Given the stochastic, dynamic nature of this system, an ADP approach may prove valuable to study this problem, although it is worth noting that Hoecherl et al. [47] showed minimal improvement over benchmark policies.

The widest application of ADP to date is by far within force employment, including: planning operations, such as airlift [18]; conducting operations, such as medical evacuation [59]; and assessing operations [48]. Though this domain has received the most attention, further potential applications exist, including: extending recent studies regarding combat medical evacuation to larger operations, such as the evacuation of passengers from a cruise ship as described in [91] or other military support to mass casualty incidents [92]; drug interdiction operations [93]; and balancing the mix between military and contract airlift to support both operations and exercises [94].

6. Conclusion

In this article, applications of ADP to support military and civilian decision makers within defence were reviewed. This review revealed a relatively small number of published articles, 18 in total, however the studies spanned the spectrum of force development, generation, and employment. The overwhelming majority of articles reviewed focused on generating policies to be used within force employment, with recent applications including combat medical evacuation, missile defence, and inventory routing. In addition, this review showed that ADP-generated policies have demonstrated a wide range of improvement over those currently practiced and myopic policies, ranging from approximately 2% to nearly 350%.

Based on these reviews, five topics related to ADP-generated policies and their deployment were discussed: classes of problems studied, evaluation, incremental versus comprehensive policy changes, policy robustness within scenarios, and additional topics in which ADP may provide benefit. All articles reviewed studied state-dependent problems, with the majority using an objective based on a cumulative contribution rather than a final contribution. Regarding evaluation, it was suggested that a currently practiced policy should be used as a benchmark if one exists in an effort to demonstrate the potential benefits to modifying existing doctrine. In the case where a currently practiced policy does not exist, several options—myopic, industry standard, etc., were suggested. Next, while proposing comprehensive policy changes may produce a larger improvement, it was recommended that incremental policy changes be sought first as this may lead to more effective interactions with the practitioner community and increase the likelihood of policies being deployed into the field. Following the trend of recent studies, it was proposed that the impact of varying scenario parameters be incorporated during policy evaluation in an effort to gauge when an policy's effectiveness degrades and an alternative policy may be required. Lastly, several sequential decision problems throughout defence, such as enterprise-wide investment planning, joint exercises and training, and mass evacuation in support of civilian authorities, to which ADP have not yet been applied and may provide benefit were highlighted.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Hausman W. Sequential decision problems: A model to exploit existing forecasters. Manage Sci 1969;16:B93–111.
[2] Puterman M. Markov decision processes: Discrete stochastic dynamic programming. Wiley series in probability and statistics, Hoboken, New Jersey: John Wiley & Son, Inc.; 2005.
[3] Powell W. Approximate dynamic programming: Solving the curses of dimensionality. 2nd ed. Hoboken, New Jersey: Wiley; 2011.
[4] Bellman R. A markovian decision process. J Math Mech 1957a;6:670–84.
[5] Bellman R. Dynamic programming. Princeton University Press; 1957b.
[6] Anderson E, Chen Y-M. A decision support system for the procurement of military equipment. Nav Res Logist 1988;35:619–32.
[7] McGinnis M, Fernandez-Gaucherand E. A dynamic programming model for the initial entry training program of the United States Army. In: Proceedings of 33rd IEEE conference on decision and control, vol. 4. 1994. p. 3632–3.
[8] Pecht E, Tishler A, Weingold N. On the choice of multi-task R & D defence projects: A case study of the Israeli missle defence system. Defence Peace Econ 2013;24:429–48.
[9] Zais M, Zhang D. A markov chain model of military personnel dynamics. Int J Prod Res 2016;54:1863–85.
[10] Keneally S, Robbins M, Lunday B. A markov decision process model for the optimal dispatch of military medical evacuation assets. Health Care Manag Sci 2016;19:111–29.
[11] Kuo F, Sloan I. Lifting the curse of dimensionality. Notices Amer Math Soc 2005;52:1320–8.
[12] Bertsekas D, Tsitsiklis J. Neuro-dynamic programming. 1st ed. Belmont, Massachusetts: Athena Scientific; 1996.
[13] Sutton R, Barto A. Reinforcement learning: An introduction. 2nd ed. Cambridge, Massachusetts: MIT Press; 2018.
[14] Bellman R, Dreyfus S. Functional approximations and dynamic programming. In: Mathematial tables and other aides to computation. Vol. 13, 1959, p. 247–51.
[15] Powell W. Perspectives of approximate dynamic programming. Ann Oper Res 2016;241:319–465.
[16] Birge J, Louveaux F. Introduction to stochastic programming. Springer series in operations research and financial engineering, 2nd ed. New York: Springer; 2011.
[17] Powell W. The optimizing-simulator: Merging simulation and optimization using approximate dynamic programming. In: Proceedings of the winter simulation conference, 2005. p. 96–109.
[18] Powell W, Bouzaiene-Ayari B, Berger J, Boukhtouta A, George A. The effect of robust decisions on the cost of uncertainty in military airlift operations. ACM Trans Model Comput Simul 2011;22.
[19] Bertsekas D. Reinforcement learning and optimal control. 1st ed. Belmon, Massachusetts: Athena Scientific; 2019.
[20] Powell W. Reinforcement learning and stochastic optimization: A unified framework for sequential decisions. 2021, https://castlelab.princeton.edu/RLSO/, (Accessed 23 August 2021).
[21] Powell W, Marar A, Gelfand J, Bowers S. Implementing real-time optimization models: A case application from the motor carrier industry. Oper Res 2002;50:571–81.
[22] Powell W. Real-time dispatching for truckload motor carriers. CRC Press; 2007, p. 15–1–20.
[23] Simão H, George A, Powell W, Gifford T, Nienow J, Day J. Approximate dynamic programming captures fleet operations for Schneider National. INFORMS J Appl Anal 2010;40:342–52.
[24] Simão H, Day J, George A, Gifford T, Nienow J, Powell W. An approximate dynamic programming algorithm for large-scale fleet management: A case application. Transp Sci 2009;43:178–97.
[25] Powell W, Bouzaiene-Ayari B, Lawrence C, Cheng C, Das S, Fiorillo R. Locomotive planning at Norfolk Southern: An optimizing simulator using approximate dynamic programming. INFORMS J Appl Anal 2014;44:567–78.
[26] Simão H, Powell W. Approximate dynamic programming for management of high-value spare parts. J Manuf Technol Manag 2008;20:147–60.
[27] Schramm H, Nicholas P, Robinson A. INFORMS editor's cut: Military O.R.. 2019, https://pubsonline.informs.org/editorscut/military.
[28] Jiang D, Powell W. Risk-averse approximate dynamic programming with quantile-based risk measures. Math Oper Res 2018;43:554–79.
[29] Powell W. Clearing the jungle of stochastic optimization. In: INFORMS TutORials in operations research. Institute for Operations Research and the Management Sciences; 2014, p. 109–37.
[30] Powell W. What you should know about approximate dynamic programming. Nav Res Logist 2009;56:239–49.
[31] Mes M, Rivera A. Approximate dynamic programming by practical examples. In: Boucherie R, van Dijk N, editors. Markov decision processes in practice. Cham: Springer International Publishing; 2017, p. 63–101.
[32] McKenna R, Robbins M, Lunday B, McCormack I. Approximate dynamic programming for the military inventory routing problem. Ann Oper Res 2020;288:391–416.
[33] Hastie T, Tibshirani R, Friedman J. The elements of statistical learning. Springer series in statistics, 2nd ed. New York: Springer; 2017, 12th printing.
[34] Powell W. A unified framework for stochastic optimization. European J Oper Res 2019;275:795–821.
[35] Spall J. Introduction to stochastic search and optimization: Estimation, simulation, and control. Wiley-interscience series in discrete mathematics and optimization, Hoboken, New Jersey: John Wiley & Sons, Ltd.; 2003.
[36] Bhatnagar S, Prasad H, Prashanth L. Stochastic recursive algorithms for optimization: Simultaneous perturbation methods. London: Springer; 2013.
[37] Bertsekas D. Approximate policy iteration: a survey and some new methods. J Control Theory Appl 2011;9:310–35.
[38] Geist M, Pietquin O. Algorithmic survey of parametric value function approximation. IEEE Trans Neural Netw Learn Syst 2013;24:845–67.
[39] Bradtke S, Barto A. Linear least-squares algorithms for temporal difference learning. Mach Learn 1996;22:33–57.
[40] Nedić A, Bertsekas D. Least squares policy evaluation algorithms with linear function approximation. Discrete Event Dyn Syst 2003;13:79–110.
[41] Bethke B, How JP, Ozdaglar A. Approximate dynamic programming using support vector regression. In: 2008 47th IEEE conference on decision and control, 2008. p. 3811–6.
[42] George A, Powell W. Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. Mach Learn 2006;65:167–98.
[43] Barr R, Golden B, Kelly J, Resende M, Stewart W. Designing and reporting on computational experiments with heuristic methods. J Heuristics 1995;1:9–32.
[44] Flint M, Fernandez E, Kelton W. Simulation analysis for UAV search algorithm design using approximate dynamic programming. Military Oper Res 2009;14:41–50.
[45] Fisher B, Brimberg J, Hurley W. An approximate dynamic programming heuristic to support non-strategic project selection for the Royal Canadian Navy. J Defense Model Simul 2015;12:83–90.
[46] Southerland J, Loerch A. Using approximate dynamic programming to model military force mix adaptation. Military Oper Res 2018;23:25–36.
[47] Hoecherl JC, Robbins MJ, Hill RR, Ahner DK. Approximate dynamic programming algorithms for United States Air Force officer sustainment. In: 2016 winter simulation conference (WSC), 2016. p. 3075–86.
[48] Ross K, Chaney R, Patek S. Neuro-dynamic programming for adaptive control of bayesian networks for global awareness. In: 1998 IEEE information technology conference, information environment for the future (Cat. No. 98EX228), 1998. p. 10–3.
[49] Bertsekas D, Homer M, Logan D, Patek S, Sandell N. Missile defense and interceptor allocation by neuro-dynamic programming. IEEE Trans Syst Man Cybern 2000;30:42–51.
[50] Popken D, Cox L. A simulation–optimization approach to air warfare planning. J Defense Model Simul 2004;1:127–40.

[51] Sztykgold A, Coppin G, Hudry O. Dynamic optimization of the strength ratio during a terrestrial conflict. In: 2007 IEEE international symposium on approximate dynamic programming and reinforcement learning, 2007. p. 241–6.
[52] Wu T, Powell W, Whisman A. The optimizing-simulator: An illustration using the military airlift problem. ACM Trans Model Comput Simul 2009;19.
[53] Ahner D, Parson C. Weapon tradeoff analysis using dynamic programming for a dynamic weapon target assignment problem within a simulation. In: Proceedings of the 2013 winter simulation conference: Simulation: Making decisions in a complex world. WSC '13, IEEE Press; 2013, p. 2831–41.
[54] Rettke A, Robbins M, Lunday B. Approximate dynamic programming for the dispatch of military medical evacuation assets. European J Oper Res 2016;254:824–39.
[55] Davis M, Robbins M, Lunday B. Approximate dynamic programming for missile defence interceptor fire control. European J Oper Res 2017;259:873–86.
[56] Laan C, Barros A, Boucherie R, Monsuur H. In: Monsuur H, Jansen J, Marchal F, editors. Security games with restricted strategies: An approximate dynamic programming approach. NL ARMS Netherlands annual review of military studies 2018: Coastal border control: From data and tasks to deployment and law enforcement, The Hague: T.M.C. Asser Press; 2018, p. 171–91.
[57] Robbins M, Jenkins P, Bastian N, Lunday B. Approximate dynamic programming for the aeromedical evacuation dispatching problem: Value function approximation utilizing multiple level aggregation. Omega 2020;91:102020.
[58] Summers D, Robbins M, Lunday B. An approximate dynamic programming approach for comparing firing policies in a networked air defence environment. Comput Oper Res 2020;117:104890.
[59] Jenkins P, Robbins M, Lunday B. Approximate dynamic programming for the military aeromedical evacuation dispatching, preemption-rerouting, and redeployment problem. European J Oper Res 2021a;290:132–43.
[60] Jenkins P, Robbins M, Lunday B. Approximate dynamic programming for military medical evacuation dispatching policies. INFORMS J Comput 2021b;33:2–26.
[61] Coifman R, Maggioni M. Diffusion wavelets. Appl Comput Harmon Anal 2006;21:53–94.
[62] Balakrishna P. Scalable approximate dynamic programming models with applications in air transportation (Ph.D. thesis), Fairfax, Virginia: George Mason University; 2009.
[63] Southerland J. Using approximate dynamic programming to adapt a military force mix (Ph.D. thesis), Fairfax, Virginia: George Mason University; 2017.
[64] Godfrey G, Powell W. An adaptive dynamic programming algorithm for dynamic fleet management, I: Single period travel times. Transp Sci 2002;36:21–39.
[65] Bradshaw A. United States Air Force officer manpower planning problem via approximate dynamic programming (M.Sc. thesis), Ohio: Air Force Institute of Technology, Wright-Patterson Air Force Base; 2016.
[66] West K. Approximate dynamic programming for the United States Air Force officer manpower planning program (M.Sc. thesis), Ohio: Air Force Institute of Technology, Wright-Patterson Air Force Base; 2017.
[67] Situ J. An approximate dynamic programming appraoch to analyzing military personnel end-strength planning (Ph.D. thesis), Fairfax, Virginia: Systems Engineering and Operations Research, George Mason University; 2018.
[68] Powell W, Ruszczyński A, Topaloglu H. Learning algorithms for separable approximations of discrete stochastic optimization problems. Math Oper Res 2004;29:814–36.
[69] Salgado E. Using approximate dynamic programming to solve the stochastic demand military inventory routing problem with direct delivery (M.Sc. thesis), Ohio: Air Force Institute of Technology, Wright-Patterson Air Force Base; 2016.
[70] Government of Canada. TERMIUM plus. 2021, https://www.btb.termiumplus.gc.ca/tpv2alpha/alpha-eng.html?lang=eng&i=1&index=alt&codom2nd_wet=1.
[71] Brown G, Dell R, Newman A. Optimizing military capital planning. Interfaces 2004;34:415–25.
[72] Rempel M, Young C. VIPOR: A visual analytics decision support tool for capital investment planning. Scientific Report DRDC-RDDC-2017-R129, Ottawa, Canada: Defence Research and Development Canada; 2017, https://cradpdf.drdc-rddc.gc.ca/PDFS/unc290/p805944_A1b.pdf.
[73] Harrison K, Elsayed S, Garanovich I, Weir T, Galister M, Boswell S, et al. Portfolio optimization for defence applications. IEEE Access 2020;8:60152–78.
[74] Gallo A. Understanding military doctrinal change during peacetime. (Ph.D. thesis), New York: Graduate School of Arts and Science, Columbia University; 2018.
[75] Sacco W, Navin D, Fiedler K, Waddell I.I. R, Long W, Buckman Jr. R. Precise formulation and evidence-based application of resource-constrained triage. Acad Emerg Med 2005;12:759–70.
[76] Saran C. AI struggles with data silos and executive misconceptions. 2019, ComputerWeekly. https://www.computerweekly.com/news/252464222/AI-struggles-with-data-silos-and-executive-misconceptions, (Accessed: 19 April 2021).
[77] Bakhshi N, Khera A, Bilato A. Navigate data management challenges to enable AI initiatives, Deloitte. 2020, https://www2.deloitte.com/content/dam/Deloitte/nl/Documents/strategy-analytics-and-ma/deloitte-nl-strategy-analytics-dail-sense-whitepaper.pdf, (Accessed 27 April 2021).
[78] Scott W. Why data silos are bad for business, Forbes. 2021, https://www.forbes.com/sites/forbestechcouncil/2018/11/19/why-data-silos-are-bad-for-business/?sh=2d0ff8765faf, (Accessed 23 April 2021).
[79] Teeple N, Dean R. NORAD modernization. CDA Institute; 2020, https://cdainstitute.ca/norad-modernization-report-three-jadc2-jado/.(Accessed 23 April 2021).
[80] Walker W, Rahman S, Cave J. Adaptive policies, policy analysis, and policy-making. European J Oper Res 2001;128:282–9.
[81] Stasko T, Gao H. Developing green fleet management strategies: Repair/retrofit/replacement decisions under environmental regulation. Transp Res A 2012;46:1216–26.
[82] Abdul-Malak D, Kharoufeh J. Optimally replacing multiple systems in a shared environment. Probab Engrg Inform Sci 2018;32:179–206.
[83] Sadeghpour H, Tavakoli A, Kazemi M, Pooya A. A novel approximate dynamic programming approach for constrained equipment replacement problems: A case study. Adv Prod Eng Manag 2019;14:355–66.
[84] Fang J, Zhao L, Fransoo JC, Van Woensel T. Sourcing strategies in supply risk management: An approximate dynamic programming approach. Comput Oper Res 2013;40:1371–82.
[85] Geng Y, Klabjan D. Approximate dynamic programming based approaches for green supply chain design. Technical Report, Industrial Engineering and Management Sciences, Northwestern University; 2014.
[86] Ghanmi A. A stochastic model for military air-to-ground munitions demand forecasting. In: 2016 3rd international conference on logistics operations management (GOL), 2016. p. 1–8.
[87] Nozhati S, Ellingwood BR, Chong EK. Stochastic optimal control methodologies in risk-informed community resilience planning. Struct Saf 2020;84:101920.
[88] Karamanis D. Stochastic dynamic programming methods for the portfolio selection problem. (Ph.D. thesis), London: The London School of Economics and Political Science; 2013.
[89] MacLeod M, Rempel M, Roi M. Decision support for optimal use of joint training funds in the Canadian Armed Forces. In: Evans G, Biles W, Bae K, editors. Analytics, operations, and strategic decision making in the public sector. Hershey, PA: IGI Global; 2019, p. 255–76.
[90] Séguin R. PARSim, a simulation model of the Royal Canadian Air Force (RCAF) pilot occupation: An assessment of the pilot occupation sustainability under high student production and reduced flying rates. In: Vitoriano B, Parlier G, editors. Proceedings of the international conference on operations research and enterprise systems. Lisbon, Portugal: SciTePress; 2015, p. 51–62.
[91] Hunter G, Chan J, Rempel M. Assessing the operational impact of infrastructure on Arctic operations. Scientific Report DRDC-RDDC-2021-R024, Ottawa, Canada: Defence Research and Development Canada; 2021, https://cradpdf.drdc-rddc.gc.ca/PDFS/unc356/p812844_A1b.pdf.
[92] Shin K, Lee T. Emergency medical service resource allocation in a mass casualty incident by integrating patient prioritization and hospital selection problems. IISE Trans 2020;52:1141–55.
[93] Sidoti D, Han X, Zhang L, Avvari GV, Ayala DFM, Mishra M, et al. Context-aware dynamic asset allocation for maritime interdiction operations. IEEE Trans Syst Man Cybern 2020;50:1055–73.
[94] Rempel M, Cai J, MacLeod M. Military versus contracted airlift: an approach to finding the right mix. Scientific Letter DRDC-RDDC-2021-L086, Ottawa, Canada: Defence Research and Development Canada; 2021.