Literature Review

The Weapon-Target Assignment Problem [Kline et al., 2019]

- As ↑ quantity and quality of missiles, effective allocation research emerged.
- Weapon Target Assignment (WTA) aka Missile Allocation Problem (MAP) ⇔ minimize probabiilty of a missile destroying a protected assignment
- Sometimes offense perspective OR defense perspective
- WTA → Static WTA (SWTA) or Dynamic WTA (DWTA)
- SWTA:
  input: num. of incoming missiles (targets), num. of interceptors (weapons), probabilities of destroying targets
  output: how many of each weapon type to shoot at each target
- DWTA includes time as a dimension. Two variants: two-stage and shoot-look-shoot
  - Two-stage DWTA:
    stages/input: 1. SWTA and 2. probability distribution of various kinds of targets
    output: 1. allocation of weapons and 2. how many weapons to reserve to minimize prob. of destruction
  - Shoot-kill-Shoot DWTA:
    replicates SWTA too, but enables observation of leakers: target that maybe survived the initial engagement for a subsequent engagement
    solution: allocation of weapons and reservation of weapons to rengage any leakers
- WTA is NP-Complete, so majority of solutions seek near-optimal solutions in real-time, or fast-enough solutions before the adversary reaches their goals.
- These solutions use heuristics or have exact solutions applied to variants of the WTA problem

Formulations

Notation:

- p_{ij} : the probability weaponi destroys target j
- $q_{ij} (= 1 - p_{ij})$: the probability weapon i fails to destroys target j
- V_j : the destructive value of target j

- x_{ij} : the number of weapons of type i assigned to target j
- K : the number of protected assets
- a_k : the value of asset k
- n : the number of targets
- m : the number of weapon types
- w_i : the number of weapons of type i
- c_{ij} : a cost parameter for assigning a weapon of type i to target j
- $\mathcal{F}$: the set of feasible assignments
- γ_{jk} : the probability target j destroys asset k
- s_j : the maximum number of weapons that can be assigned to target j
- t : the number of stages

SWTA's main formulation:

$$\begin{array}{rl} \min & \sum_{j=1}^{n} V_j \prod_{i=1}^{m} q_{ij}^{x_{ij}} \\ \text{s.t.} & \sum_{j=1}^{n} x_{ij} \leq w_i, \text{ for } i = 1, \ldots, m \\ & x_{ij} \in \mathbb{Z}_+, \text{for } i = 1, \ldots, m, j = 1, \ldots, n \end{array}$$

In English: find the best assignment of number of weapons, across all types, that minimizes the survival rate of the targets, with higher emphasis of those with more destructive value. The "such that (s.t.)" requriements indicate that we must respect our supply capacity of weapons and that we must allocate at least weapon of each type to all targets.

Other formulations: They make simplifying assumptions, such as assuming that the probability of using any weapon to destroy a target is the same or that there is only a capacity of one weapon of each type. Maybe there is one weapon type per target. All of these different assumptions simplify optimization in some way, but obviously each has its pros and cons in terms of optimizability and applicability to real-world scenarios.

DWTA's main formulation: - I'm not going to really elaborate on the math; it gets substantially more complicated. I think an important takeaway is that because SWTA was already intractable with its large amount of permutations to begin with, DWTA is certainly intractable as well, given how it's basically just a stack of SWTAs. - Feel free to look at the math yourself, but be willing to spend a lot of attention and time. I just did not think it was worth it. - Antoher important takeaway: it's a must to use approximation methods.

Exact Algorithms for SWTA

Maximum Marginal Return (MMR)

This algorithm assums that the probabiilty of kill for any weapon target to target j is the same.

Then, the optimal solution is:

1. Assign $x_{ij} = 1$ where $\{i, j\} \in argmax(V_j p_{ij})$
2. $V_j \leftarrow V_j(1 - p_{ij})$
3. $p(i, \cdot) \leftarrow 0$ and $p(\cdot, j) \leftarrow 0$
4. Repeat until all weapons have been assigned

Note that it's best to divide the weapons evenly across all targets when there is only one probability of kill, regardless of weapon or target.

On Solving the Original SWTA Formulation

When you try exhaustive searches: a problem with 9 weapons and 8 targets take 13 min to run to completion, and adding one additional target takes 43.7 min. to run to completion. Thus, there is a combinatorial explosion in run time as a function of the problem size.

Brief Mention of Other Algorithms

- Branch-and-bound
- Lower bounding strategies (generalized network flow, MMR, and minimum cost flow)
- Linear integer programming
- Joint Munition Effectiveness Manual (JMEM)

Exact Algorithms for DWTA

- Mathematics-based (Burr et al. (1985) & Soland (1987) & Hosein (1989))
- Concave Adaptive Value Estimation (CAVE) with modified MMR (aka MMR Plus Algorithm)
- Dynamic Programming

Heruistics for SWTA

All the above solutions focus on the optimal solution, but heuristic algorithms focus on real-time solutions.

- Genetic algorithms
- Very Large Scale Neighborhood (VLSN) search metahueristic

- Ant Colony Optimization
- Integer relaxed NLP and rounding schemes
- Neural networks
- Network-flow-based construction heruistic
- Simulated Annealing (SA)
- Variable Neighbor Search (VNS)
- Tabu Search
- Particle Swarm Optimization (PSO)
- Lagrangian relaxation Branch and Bound
- Hungarian Algorithm

Heruistics for DWTA

- ALIAS algorithm
- Decomposition algorithm
- Virtual permutations
- Rule-based
- Hungarian Algorithm
- Neuro-dynamic programming to obtain near optimal policies. Optimal policies are obtained through dynamic programming

Discussion

The branch and bound algorithm and the genetic algorithm are both widely used in the literature. Bogdanowicz (2012)'s algorithm or Xin et al. (2010)'s rule-based hueristic efficiently exploit the special structure of the problem.

Applying reinforcement learning to the weapon assignment problem in air defence [Mouton et al., 2011]

Abstract

- Monte Carlo control algorithm with Exploring Starts (MCES)
- Q-learning: an off-policy temporal-difference (TD) learning-control algorithm
- These algos. used in SIMPLIFIED version of weapon assignment (WA) problem

Introduction

- RL uses framework to define interaction between learning agent and environment in terms of states, actions, rewards
- RL agent must discover by trial and error to get highest reward
- No exact method exists for the WA problem, evne when it is realtively small-sized
- Compare to threat evaluation (TE), WA decisions are more quantifiable
- This article discusses whether RL is suitable for WA.

  Flow: RL & related work overview, greater context → command and control (C2), threat evaluation and weapon assignment (TEWA) → how WA problem was modeled and how MCES & Q-learning were applied → compare experimental results → ideas for future study

Overview of reinforcement learning and related work

- Azak & Bayrak implemented agents for TWEA problems of C2 systems for decision-making.
- They sought to optimize decision-making performance for multi-armed-platform-TEWA problems
- RL can be seen as a Markov Decision Process (MDP). This allows agent designers to work with systems in their current state, without worrying about how it came to be in that state in the first place.
- RL loop: agent given info. about env. through sensory input → action → reward
- RL algos. maps each world state to agent actions

Command and control

- C2 system = facilities, equipment, communications, procedures, personnel essential for planning, directing, and controlling for a commander.
- It involves observation, orientation, decision, and action.
- Observation: gather info. relevant to decision
- Orientation: cognitive effort; breaking down problem into sub-problems and matching each with emergency plan → overall action plan
- Decision: whether to execute the action plan
- Action: execution of chosen course of action or plan, e.g. physical attack/movement, order issuance, sensor maintenance for better future observations

- I don't think the rest of this subsection and the next one on threat evaluation and weapon assignment is that important for the lit. review.

Modeling the weapon assignment problem

- Asset to defend is at the center of the grid.
- 4 missile stations defended the asset. Once these stations were placed, they remained fixed.
- Threat takes straight line towards asset, and flies in from one of the cells. Once a threat reaches the asset, it eliminates the asset.
- Threat stays in specific cell while all the weapons get a turn to shoot at the threat. If all weapons miss, the threat moves one cell closer.
- If threat on weapon position, no shots can be fired. Thus, the threat gets a free pass to move one cell closer.
-

Optimization of Weapon-Target Pairings Based on Kill Probabilities [Bogdanowicz et al., 2013]

Kill probabilities are not additive; the odds that multiple weapons kill a target is greater than the sum of the odds that each one kills the target summed up, since you can do things such as attack from various angles. This allows us to constrain the algorithm further to improve performance. Furthermore, user-imposed constraints (such as restrictions on the number of weapons that can be assigned to a given target) can massively improve performance. Since different algorithms are better at different levels of searching (genetic algorithms are good at global searching but get stuck in local optima easily, harmony search is very good at local optimization), switching from one to another at various points in the calculation can also dramatically improve performance.

A Coordinated Air Defense Learning System Based on Immunized Classifier Systems [Nantogma et al., 2021]

Multi-agent systems can be applied to defense and security applications, specifically to solve threat evaluation and weapon assignment problems. These applications require complex systems, due to both the complexity of the application domain and incompleteness and uncertainty of perceived information. In addition, many battlefield decisions are time sensitive. All of the above make it difficult to design an effective model. This paper combines an artificial immune system and a learning classifier system to tackle common threat evaluation and weapon assignment problems.

An artificial immune system is a multi-agent, decentralized information processing system that is capable of learning and remembering. It uses computational techniques derived from defense mechanisms native to biological immune systems to accomplish its goals. In an immune system, an antigen is a harmful foreign substance, like a bacterium. B-cells, or lymphocytes, are immune cells that produce antibodies: markers that mark an antigen to be eaten by a T-cell.

- **CLONALG**, one such technique based on clonal selection, uses the idea of immunological memory to implement the learning and remembering component of the artificial immune system. Biological immune systems implement memory by cloning two types of B-cell. One type of B-cell immediately acts to combat infection, while the other type is retained by the immune system and grants immunity to future infection.
- **Danger theory** is another technique based on the fact that immune systems only respond to harmful foreign substances. Similarly, in an immunized air defense system, it's important to determine which threat to prioritize. When making a weapon target assignment, it's important to take the threat level of the targets into account.

  Whenever a target, or hostile entity is detected, a danger zone is constructed, based on the trajectory of the detected target. The value of the combat units inside this danger zone signals how much priority should be given in intercepting the target.
- **Network theory**: In an immune system, the antibodies of many different B-cells, of the same and different type, are able to recognize and interact with each other, even without the presence of foreign substances. When one antibody recognizes the other, B-cells will produce different antibodies meant to suppress the recognized antibody, while the antibody that did the recognizing will get a boost in production. The resulting network of antibody interactions serves to broaden the diversity of the existing antibody population.

A learning classifier system is a decision making system that makes use of simple if-then rules, or classifiers. As it executes, the classifiers are slowly improved. In operation, it consists of three phases: performance, reinforcement, and discovery, in that order.

- **Performance:** In the performance phase, actions are selected and executed using an existing batch of classifiers. The classifiers that have that specific action are taken and stored in an action set. The executed action has an associated reward.
- **Reinforcement:** In the reinforcement phase, the parameters in the classifiers are updated using the reward from the performance phase.

  The update equations for this phase for a classifier *cl*, are as follows:

  - **Prediction** $cl.p$**:**

    $$cl.p = cl.p + \beta(R - cl.p) \tag{1}$$

  - **Prediction Error** $cl.\epsilon$**:**

    $$cl.\epsilon = cl.\epsilon + \beta(|R - cl.p| - cl.\epsilon) \tag{2}$$

- **Fitness $cl.F$:**

$$cl.F = cl.F + \beta(\hat{\lambda}(cl) - cl.F) \tag{3}$$

- **Accuracy $\lambda(cl)$:**

$$\lambda(cl) = \begin{cases} 1 & \text{if } cl.\epsilon < \epsilon_0 \\ \alpha(\frac{cl.\epsilon}{\epsilon_0})^{-v} & \text{if } cl.\epsilon \geq \epsilon_0 \end{cases} \tag{4}$$

  Where ϵ_0 is an accuracy criterion constant. A classifier is accurate if $cl.\epsilon$ is smaller than ϵ_0. α and v are hyper-parameters used to control the rate at which the accuracy reduces.

- **Relative accuracy $\hat{\lambda}(cl)$:**

$$\hat{\lambda}(cl) = \frac{cl.n \times \lambda(cl)}{\sigma_{cl_b \in [A]} \lambda(b) \times b.n} \tag{5}$$

- **Discovery:** In the discovery phase, the algorithm applies a genetic algorithm to generate a new batch of classifiers. Two parents with high fitness are selected from the action set, and offspring are produced using a crossover algorithm and a random bit-flip mutation. Classifiers with low fitness are deleted if the number of total classifiers in the batch get too large, and classifiers whose conditionals are already included in more accurate, experienced classifiers can be deleted.

In a generic air defense mission, the two problems to solve are action strategy selection and cooperation. Thus, we use two agents, one in charge of generating strategies, and one in charge of coordinating defenses. The strategy generation agent uses a learning classifier system to learn the right weapon and amount of ammunition to use against given targets. These classifiers are passed to the strategy coordination agent. Then, the strategy coordination agent uses immune network dynamics to decide what to do, given the classifiers that the strategy generation agent created.

The strategy generation agent treats each weapon type as a B-cell, or decision unit, that each have their own classifiers, or antibodies. Each classifier will have control decisions as their consequent (the THEN component of the IF-THEN statement). Enemy forces are expressed in terms of their intent, capability, and opportunity. When individual B-cells activate on detection of antigens, or hostile units, the strategy generation agent generates a set of classifiers, from which the strategy generation agent selects the most appropriate action. The strategy generation agent interfaces with the environment through an intermediate module, which also provides local detectors for the B-cells.

The strategy coordination agent receives classifiers from the strategy generation agent, establishes connections between the classifiers, forming a network. Then, it applies immune network dynamics. Classifier i is said to be connected to classifier j if they produce antibodies for the same target. The concentrations for each classifier is calculated, and the one with the highest concentration ultimately is chosen as the action. The output classifiers from this immune network dynamics process are the ones that undergo genetic operations and updates.

Specific Equations:

- **Objective Function:** At time T, for targets detected H and C categories of weapons for each agent, with each category C_p having j ammunition, the objective function is

$$O_b = \begin{cases} min & \sum_{j=1}^{H'(t)} \sum_{i=1}^{C} \beta_i q_{ij} \delta_{ij} \\ max & \sum_{j=1}^{H'(t)} v_j (1 - \prod_{i=1}^{C} (1 - p_{ij})^{\delta_{ij}(s)}) \end{cases} \tag{6}$$

  Where H' is the remaining number of unassigned detected targets/enemy units, δ_{ij} is the minimum delay before weapon i can be deployed against target j based on ready time and current allocation of that weapon, q_{ij} is the quantity of ammunition of weapon of type i allocated to target j, b_i is the unit cost of the ammunition of weapon i, v_j is the threat value of target j, and p_{ij} is the weapon kill probability.

- **Expected Action Payoff $P(a_i)$:** Given a set of classifiers matching the attributes of a specific B-cell $[M]$, the expected action payoff for a specific action a_i is

$$P(a_i) = \frac{\sum_{cl_k \in [M] | a_i} cl_k.p \times cl_k.F * \omega}{\sum_{cl_k \in [M] | a_i} cl_l.F * \omega} \tag{7}$$

  Where ω is the affinity between the B-cells' classifiers and antigens. This is the reward that the agent receives for performing a specific action.

- **Classifier-Antigen Affinity ω:**

$$\omega = [1 - \prod_{i=1}^{q} (1 - d_g * p_g * w_e * r_t)] * v_c \tag{8}$$

  Where q is the quantity of ammo suggested by the classifier, d_g is the normalized distance between target c and the friendly unit (B-cell), p_g is the speed advantage of firing the weapon against the target, w_e is the kill probability of the weapon suggested, r_t is the ready time of the weapon if it were to be deployed, and v_c is an estimate for the value of the target.

  This value tells you how good a weapon (described in a classifier) is, against a specific antigen.

- **Classifier Concentration $a_i(t+1)$:**

$$\frac{da_i(t+1)}{dt} = \left(\alpha \sum_{j=1}^{N} m_{ji} a_j(t) - \beta \sum_{j=1}^{N} m_{ik} a_k(t) + \gamma m_i - k \right) a_i(t) \tag{9}$$

  Where N is the number of classifiers that deal with the target antigen, m_i is the affinity between classifier i and the target antigen, m_{ji} is the mutual stimulus coefficient of antibody j on classifier i, m_{ki} is the inhibitory effect of classifier k on classifier i, k is the natural death rate of classifier i, $a_i(t)$, $a_j(t)$, and $a_k(t)$ are the bounded concentrations that are imposed on the classifiers, and coefficients α, β, and γ are weight factors that determine the significance of each term.

  This ordinary differential equation embodies the immune network dynamics that this algorithm leverages. Classifiers compete with each other via this concentration value. Since it is an ordinary differential equation, you have to calculate the next step's value using previous values, like any other algorithmic differential equation solver.

The state-of-the-art review on resource allocation problem using artificial intelligence methods on various computing paradigms [Joloudari et al., 2022]

This is a comprehensive literature study on ML and deep learning methods for the resource allocation problem in different computing environments (distributed, IoT, etc.)

Interesting Resource Allocation articles and their main ideas:

- **Othman and Nayan (2019):**

  Review of solutions based on reinforcement and heuristic learning using a dynamic and adaptive allocation resource. Deep learning methods had faster, more accurate convergence.

  A. Othman and N. A. Nayan, "Efficient admission control and resource allocation mechanisms for public safety communications over 5G network slice," Telecommunication Systems, vol. 72, no. 4, pp. 595-607, 2019.

- **Yousefzai et. al. (2017):**

  Investigation of the resource allocation problem in cloud computing. Examination of different schemes based on cloud computing resources using effective features like optimization goals, optimization methods, design approaches, and useful functions.

  A. Yousafzai et al., "Cloud resource allocation schemes: review, taxonomy, and opportunities," Knowledge and Information Systems, vol. 50, no. 2, pp. 347-381, 2017.

- **Han et. al. (2018):**

  Describes an evolutionary reinforcement learning / genetic algorithm that works 90% better than the naive strategy in long term use of a network.

  B. Han, J. Lianghai, and H. D. Schotten, "Slice as an evolutionary service: Genetic optimization for inter-slice resource management in 5G networks," IEEE Access, vol. 6, pp. 33137-33147, 2018.

A Survey on Weapon Target Allocation Models [Ghanbari et al., 2021]

- Two key components of command and control are: weapon target allocation (WTA) and threat evaluation.
- Resource allocation is stochastic/uncertain with regard to the WTA problem.
- The WTA component of the WTA problem can be considered in 3 parts: response planning, response execution, outcome assessment.
- There exists three basic models:

Basic Model 1: For maximizing damage to enemy (minimize expected target values F), we have

$$\min(\ F\ ) \ = \ \sum_{i=1}^{|T|} V_i \ \Pi_{k=1}^{|W|}(1-P_{ik})^{x_ik}$$

This is the general WTA formula.

Basic Model 2: For allocation of available units to maximize expected total protection value J, we have

$$\max(\ J\ ) \ = \ \sum_{j=1}^{|A|} \omega_j \ \Pi_{i\in G_j}(1-\pi_{ij}\Pi_{k=1}^{|W|}(1-P_{ik})^{x_ik}$$

Basic Model 3: This is the model for Dynamic WTA at stage t given the total expected combat value of surviving assets

$$\max(\ J_tX^t\ ) \ = \ \sum_{j=1}^{|A(t)|} \omega_j \ \Pi_{i=1}^{|T(t)|} \left[1-\pi_{ij}\Pi_{h=t}^{S}\Pi_{k=1}^{|W(t)|}(1-p_{ik}(h))^{x_{ih}(h)}\right]$$

Variable Definitions	
Sets	
T_i	Set of detected threats $i = 1, 2, \cdots, I$.
w_k	Set of resources $k = 1, 2, \cdots, K$.
A_j	Set of assets $j = 1, 2, \cdots, J$.
S	Set of engagement stages, $s = 1, 2, \cdots, S$.
$A(t), T(t), W(t)$	Set of current ”defended assets, hostile targets, and available weapons during stage t, respectively.”
Parameters	
P_{ik}	Estimated effectiveness/probability that weapon $w_k \in W$ neutralizes threat $T_i \in T$ if assigned to it.
π_{ij}	Estimated probability threat $T_i \in T$ destroys asset $A_j \in A$.
V_{ik}	Threat value of the threat-asset pair (T_i, A_j).
ω_j	Protection value of asset A_j.
C_{ik}	Resource usage cost for assigning w_k to T_i.
Variables	
X_{ik}	Is 1 if resource w_k is assigned to T_i, 0 otherwise.
$[X^s_{ik}]_{I \times K}$	Decision matrix at stage s.
h	Index of stages $t, \cdots, S$.

- Dynamic WTA (DWTA) suffer from curse of dimensionality.

- WTA problem has two perspectives: *single platform perspective* and *force coordination perspective*. The former is single platform defending one asset against incoming threats, the latter is a command and control platform defending multiple assets.

- Within these perspectives exist two paradigms: *threat-by-threat* and *multi-threat*. The former being sequential targeting and the latter being parallel targeting.

- There also exists two different prioritizations of defense, as shown by basic models 1 and 2.
- Static WTA (SWTA) constraints: $X_{ik} \in \{0,1\} \forall i \in \{1,2,\cdots,|T|\}, \forall k \in \{1,2,\cdots,|W|\}$ given the equations: $\sum_{i=1}^{|T|} X_{ik} = 1 \quad \forall k \in \{1,2,\cdots,|W|\}$ if each firing unit must be assigned a target and $\sum_{i=1}^{|T|} X_{ik} \leqslant 1 \quad \forall k \in \{1,2,\cdots,|W|\}$ otherwise, with X_{ik} being a target i being assigned to a resource k in the constraint matrix X.

- Dynamic WTA (DWTA) problems have more constraints as follows:

Weapon multi-target constraint: This constraint describes multi-target systems. As each multi-target system can also be considered as separate systems, $n_k = 1 \forall k \in \{1,2,\cdots,W\}$.

$$\sum_{i=1}^{|T|} x_{ik}(t) \leqslant n_k \quad \forall t \in \{1,2,\cdots,S\}, \forall k \in \{1,2,\cdots,|W|\}$$

Strategy constraint: This constraint limits system-usage cost per target at stage t. m_i depends on performance of available resource k on target i. For missile systems, $m_i = 1$, and for artillery systems, $m_1 \geqslant 1$.

$$\sum_{k=1}^{|W|} x_{ik}(t) \leqslant m_i \quad \forall t \in \{1,2,\cdots,S\}, \forall i \in \{1,2,\cdots,|T|\}$$

Resource constraint: This constraint governs over ammunition availability.

$$\sum_{t=1}^{S} \sum_{i=1}^{|T|} x_{ik}(t) \leqslant N_k, \quad \forall k \in \{1,2,\cdots,|W|\}$$

Engagement feasibility constraint: This constraint is over the resource-target relationship: if a target i can be hit by a resource k at stage t, then $f_{ik}(t) = 1$, and $f_{ik}(t) = 0$ otherwise.

$$x_{ik}(t) \leqslant f_{ik}(t), \qquad \forall t \in \{1,2,\cdots,S\}, \forall i \in \{1,2,\cdots,|T|\}$$

An approximate dynamic programming approach for comparing firing policies in a networked air defense environment [Summers et al., 2020]

- For the interception of theater ballistic missiles (TBMs), the U.S. Air Force noted that there exists two options for the WTAP: the Markov decision processes, an extension of the Markov chain, and approximate dynamic programming (ADP) for WTAPs involving TBMs.

- For the latter option, there exist two different algorithmic approaches: approximate value iteration and approximate policy iteration (API); the paper uses the latter.
- The API algorithmic strategy maps the system state – includes incoming target amount, current asset health, and interceptor health –
  to reaction fire against incoming targets, specifically how many interceptors to assign to each incoming target.
- MDP Formula:
  Let $\Gamma = \{1, 2, \cdots, T\}$, $T \leqslant \infty$, where the number of decision epochs T is random and follows a geometric distribution with parameter $0 \leqslant \gamma < 1$.
  Asset status component $a_t = (a_{ti})_{i \in A} \equiv (a_{t1}, a_{t2}, \cdots, a_{t|A|}$, where set of all assets $A = \{1, 2, \cdots, |A|\}$ and health of asset $a_{ti} \in \{0, 0.25, 0.5, 0.75, 1\}$, with 0 being destroyed and 1 being undamaged.
  Let resource inventory component $R_t = (R_{ti})_{i \in A} \equiv (R_{t1}, R_{t2}, \cdots, R_{t|A|})$, with status element $R_{ti} \in \{0, 1, \cdots r_i\}$, with
- The MDP formula is too complex to be summarized, please refer to pages 7-10 of the paper.
- There are two value function approximations for API: least squares policy evaluation (LSPE) and least squares temporal difference (LSTD).

API-LSPE Algorithm

```
Step 0: Initialize $\theta^0$ .
Step 1:
        for $n = 1$ to $N$ (Policy Improvement Loop).
    Step 2:
            for $k = 1$ to $K$ (Policy Evaluation Loop).
              i. Generate a random post-decision state $S^x_{t-1,k}$ .
             ii. Record basis function evaluation $\phi(S^x_{t-1,k})$ .
            iii. Simulate transition to next pre-decision state $S_{t,k}$ using equation (6).
             iv. Determine decision $x_{t,k} = X^{\pi_{adp}}(S_{t,k}|\theta^{n-1})$ using equations (5), (7), (9).
              v. Record cost $C(S_{t,k}, x_{t,k})$ .
            end for
            Update $\theta^n$ and policy:
            $\hat{\theta} = [(\Phi_{t-1})^T(\Phi_{t-1}]^-1(\Phi_{t-1})^T C_t$
            $\theta^n = a_n\hat{\theta} + (1 - a_n)\theta^{n-1}$
        end for
        Return $X^{\pi_{adp}}(\cdot|\theta^N)$ and $\theta^N$
        End.
```

API-LSTD Algorithm

```
Step 0: Initialize $\theta^0$ .
Step 1:
        for $n = 1$ to $N$ (Policy Improvement Loop)
    Step 2:
            for $k = 1$ to $K$ (Policy Evaluation Loop).
              i. Generate a random post-decision state $S^x_{t-1,k}$ .
             ii. Record basis function evaluation $\phi(S^x_{t-1,k})$ .
            iii. Simulate transition to next pre-decision state $S_{t,k}$ using equation (6).
             iv. Determine decision $x_{t,k} = X^{\pi_{adp}}(S_{t,k}|\theta^{n-1})$ using equations (5), (7), (9).
              v. Record cost $C(S_{t,k}, x_{t,k})$ .
             vi. Record next post-decision state $S^x_{t,k}$ with $x_{t,k}$ and equation (5).
            vii. Record basis function evaluation $\phi(S^x_{t,k})$ .
            end for
            Update $\theta^n$ and policy:
            $\hat{\theta} = [(\Phi_{t-1} - \gamma\Phi_t)^T(\Phi_{t-1} - \gamma\Phi_t]^{-}1(\Phi_{t-1} - \gamma\Phi_t)^T C_t$
            $\theta^n = a_n\hat{\theta} + (1 - a_n)\theta^{n-1}$
        end for
        Return $X^{\pi_{adp}}(\cdot|\theta^N)$ and $\theta^N$
        End.
```

With equations $5, 6, 7, 9$ being expressed in page 10-11.

- θ is a parameter vector created from the set of basis functions $\phi_f(S_t))_{f\in F}$, with F being the set of basis functions that reduces the size of the state variable to the most significant factors.
- We also have the basic function vector $\Phi_{t-1} \rightarrow \begin{bmatrix} \phi(S_{t-1,1}) \\ \vdots \\ \phi(S^x_{t-1\,K}) \end{bmatrix}$ and cost vector $C_t \rightarrow \begin{bmatrix} C(S_{t,1}) \\ \vdots \\ C(S_{t,K}) \end{bmatrix}$.
- We can estimate θ with $a_n = \frac{a}{a+n-1}$, $a \in (0, \infty)$
- Of the two ADP methods and given the standard methods of engagement – them being the one-target-per-interceptor, or ”Match” Policy, and the one-target-per-two-interceptors, or ”Overmatch” Policy – the LSPE algorithm outperforms the standard methods of engaging TBMs when the duration of the engagement is inherently short or the targets have high hit likelihood – i.e. are high quality. However, the Match

Policy outperforms the LSPE algorithm if the engagement is long and the attacker's weapons are of lower quality. The LSTD algorithm follows this same trend.

References

[Bogdanowicz et al., 2013] Bogdanowicz, Z. R., Tolano, A., Patel, K., and Coleman, N. P. (2013). Optimization of weapon-target pairings based on kill probabilities. *IEEE Transactions on Cybernetics*, 43(6):1835–1844.

[Ghanbari et al., 2021] Ghanbari, A. A., Mohammadnia, M., Sadatinejad, S. A., and Alaei, H. (2021). A survey on weapon target allocation models and applications. *Computational Optimization Techniques and Applications*.

[Joloudari et al., 2022] Joloudari, J. H., Mojrian, S., Saadatfar, H., Nodehi, I., Fazl, F., shirkharkolaie, S. K., Alizadehsani, R., Kabir, H. M. D., Tan, R.-S., and Acharya, U. R. (2022). The state-of-the-art review on resource allocation problem using artificial intelligence methods on various computing paradigms.

[Kline et al., 2019] Kline, A., Ahner, D., and Hill, R. (2019). The weapon-target assignment problem. *Computers & Operations Research*, 105:226–236.

[Mouton et al., 2011] Mouton, H., Roodt, J., and le Roux, H. (2011). Applying reinforcement learning to the weapon assignment problem in air defence. *Scientia Militaria: South African Journal of Military Studies*, 39:99–116.

[Nantogma et al., 2021] Nantogma, S., Xu, Y., and Ran, W. (2021). A coordinated air defense learning system based on immunized classifier systems. *Symmetry*, 13(2):271.

[Summers et al., 2020] Summers, D. S., Robbins, M. J., and Lunday, B. J. (2020). An approximate dynamic programming approach for comparing firing policies in a networked air defense environment. *Comput. Oper. Res.*, 117:104890.