Target Assignment Strategy Using Reinforcement Learning

Prasuna Saka
Scientist, Advanced Systems Laboratory
DRDO, Min. of Defence
Hyderabad, India
sprasuna.asl@gmail.com

Ashwin Chandar O.R.
Scientist, Advanced Systems Laboratory
DRDO, Min. of Defence
Hyderabad, India
orashwin.asl@gmail.com

Bala Raju G.
Scientist, Advanced Systems Laboratory
DRDO, Min. of Defence
Hyderabad, India
balarajug.asl@gmail.com

doi: https://doi.org/10.37285/bsp.SACAD2025.04

***Abstract*—Target Selection Strategy in War Game Planning is aimed at selecting an optimal Target set among the available list of Targets which maximises the end mission objectives. Recent wars has clearly proven that with the sophisticated air defence systems in place, the hit probability of the adversary Targets is very low. To really hard hit a High Value Target, multiple weapons should be aimed upon a single Target, which eventually has a direct implication on the cost efficacy factor of War Game Planning. Hence, in high heat environment of war scenario, selecting the most favourable Target list which maximises the outcome of an operation is of utmost importance. In this paper, Reinforcement Learning (RL), a machine learning based approach is proposed to find an optimal solution in determining the appropriate Targets. The problem was showcased as an equivalent to multi-armed bandit problem which is the simplest form of RL. Multi-armed bandit (MAB) solutions employ variants of simple greedy algorithms to find an optimal long term reward. In addition, limitations of MAB are addressed by considering Upper Confidence Based (UCB) solution strategy. The efficacy of the proposed approach is illustrated by applying both MAB and UCB to an illustrative case study. At the end, it is shown that RL is a potential solution to this problem using several indicative scenarios.**

I. Introduction

It is very evident from the modern warfare that air missile defence systems are dictating the entire cycle of war game strategy. War game strategy is a complex process involving following four major steps

- Targeting and Selection - Identification and Prioritisation of the Target
- Weapon Selection and Planning - Choosing an appropriate weapon and developing a plan to deploy the weapon
- Execution and Delivery - Physical Movement of the weapon and Target Engagement
- Assessment and Feedback - Evaluation of the result of Target engagement and a plan for future refinements

The first two steps combined together is coined as Weapon Target Assignment (WTA) problem [1], [2], [3], [4]. This deeply involves with the identification of High Value Prioritised Target list usually termed as Target Selection and followed by an appropriate Weapon Assignment Strategy.

In military operations research, Weapon Target Assignment (WTA) problem is considered as a classical optimization challenge, which deals with the optimal assignment of weapons to targets in order to maximize the efficiency of defense or attack operations. The end objective of WTA problem is to find an optimal strategy such that Target Selection and Weapon Assignment to Targets results in either maximizing the destruction of enemy assets or minimizing the survival of threats. This problem is classified as NP-complete and finds applications in air defense systems, missile interception, and autonomous weapon planning. Let there be m weapons and n targets. Each weapon w_i has a probability p_{ij} of destroying target t_j. Each target t_j has a value or importance v_j. Moreover, with air defence systems in position, it is not easy to hit a target. As a result, each target in turn is associated with a kill probability p_j. Now, the goal is to select appropriate target list and assign weapons to targets such that the total reward value of a mission is maximised. Due to its combinatorial nature, the WTA problem is computationally expensive. The following methods are commonly employed:

A. Exact Algorithms

- Integer Linear Programming (ILP)
- Branch and Bound

B. Heuristic and Meta-heuristic Approaches

- Genetic Algorithms (GA)
- Simulated Annealing (SA)
- Particle Swarm Optimization (PSO)
- Ant Colony Optimization (ACO)

C. Greedy Approaches

Heuristics that assign the best available weapon to the highest-value target with the highest kill probability.

The variants in approaches include

- **Static vs. Dynamic:** All assignments made at once versus over time.
- **Homogeneous vs. Heterogeneous Weapons:** Same or different weapon capabilities.
- **Multiple Weapons per Target:** Targets may require multiple hits for destruction.

Target Selection together with Weapon Assignment strategy is the core of the War game planning which plays a significant role in achieving the end objectives of a mission. In any warfare, attackers objective is to inflict maximum damage to the adversary assets. At the end of any warfare, the amount of damage inflicted to the adversary Targets decides the accomplishment of the mission. The amount of damage caused to the adversaries depends not only on the number of targets devastated by an attackers weapon system but also on the value of the asset targeted by a weapon. This means, a single high value target may be equal to neutralising a

number of low valued targets. Hence, a two pronged approach need to be adopted to maximise the end objective as follows

- Firstly, appropriate target selection taking into account of the importance of the value of the Target
- Secondly, appropriate selection of a weapon with high hit probability i.e a weapon with a low kill detection by enemy air defence system.

It is imperative that the High Value Targets in general are shielded by a strong defensive system. With the improved efficacy of air defence systems, augmented with uncertainty in estimating the capability of the adversaries air defence systems, *Target Selection* has become a sophisticated problem. Under such scenario, even with a weapon of high hit probability, damage potential imparted to the adversary Targets is uncertain, making inefficient use of the weapon. Here comes, the role of intelligent computational algorithms which aids in better decision making paradigm under pressure. Since decision making strategy involves a parameter of uncertainty, learning algorithms are a better choice. Here, in this paper a Reinforcement learning based strategy in Target Selection Problem is proposed, taking into consideration of the uncertainties associated with the hit probability of a Target in order to maximise the objectives of a mission.

II. Literature Survey

A significant amount of work is deliberated on the optimal allocation of weapons to the targets to maximise the objectives of a mission. A comprehensive survey in this area of work is provided by an author in his article [10]. The author in his publication provides a thorough review of existing work on WTA, with an end focus on the possible future directions in addressing the the WTA problem, based on the learnings made from the content covered in the literature review. However, with the fast pace technology advancements, weapon systems have undergone incessant updates, leading to the diversity of aircraft and missile types, the range of threats they face, complex interception logics, all phases of War Gaming Strategy has become exceptionally a complex problem.

Basically, the problem of Weapon Target Assignment is required to be done in two situations - Attacking mode or Defending mode. Most of the research work in addressing the WTA problem is carried out in defending mode. Many authors [8], [11], [9], [12] dealt the problem of appropriate weapon assignment to the incoming lethal targets in a dynamic situation using various machine learning paradigms. The authors in [9], gives a detailed picture of using RL based approach for Threat Evaluation and Weapon assignment (TEWA). In his paper, he addresses the TEWA problem by assessing the threat in real time, followed by assignment of a tactical weapon against aerial threats to protect ground-based defended assets (DAs). Also, the author [8] in his study, focuses on the one-on-many Dynamic WTA (one asset versus multiple threats), aiming to maximize the assets survivability. The authors addresses the DWTA problem by leveraging Deep Reinforcement Learning (DRL) and Graph Neural networks (GNN) AI based methodology. The author in [7] uses the same methodology in case of Hypersonic strike weaponry systems. Here, in this paper we try to address the problem of Target Selection and Prioritisation when a user is in attackers mode.

III. Problem Formulation

During the Target Selection Phase of War Gaming Strategy, the attacker has to choose m Targets among the available list of n Targets, where $n > m$. In general, not all Targets are equally important, hence, every Target is in-turn associated with a value which is an indicative measure of the worthiness of choosing that particular Target. Let the word *Figure of Merit*, denoted as *Fm* indicates the worthiness of the Target. Given the figure of merit of the Targets in terms of its importance to the attacker, it becomes trivial to choose those Targets whose figure of merit is on the higher side. However, the problem is aggravated by the fact, there is an uncertainty in the kill probability of the Target under consideration.
This uncertainty arises due to two reasons -

- Primarily, there always lies an uncertainty in the hit probability of a Target by a weapon due to the limitations of the weapon system itself and
- Secondarily, High Value Targets are usually shielded by defensive systems.

It is important to understand that the uncertainty factor due to the primary reason has to be addressed by the attacker itself. Having a high reliable, low detectability weapons can reduce this uncertainty. Coming to the uncertainty factor with respect to the secondary reason, having an high Intelligence Information System which provides the data facts about lethal targets and protective measures in place for such targets can aid the attacker in selecting a Target. However, any amount of information in this regard is not fully predictive, their always lies an uncertainty factor. This uncertainty factor is a direct measure of the hit probability of the Target. It is important to note that the information about the Targets known to the attacker are just estimates but may not represent actual values. These estimates are made based on the subjective knowledge gained by the commander in force which is backed up by past beliefs, operational environment, military situation awareness and many more. Based on the available information an estimate on the hit probability of the target can be arrived upon. Eventually, given the Target list along with their figure of merit associated with hit probability, the problem can be formulated in the following way

- Problem Formulation

Given:

- A set of Targets, *T1, T2, ... Tn*
- Figure of Merit for each Target, *FM1, FM2 , ... FMn*
- Hit probability of each Target, *P1, P2, ... Pn*

Assumptions:

- Out of available n number of targets, m number of targets to be chosen, where $n > m$
- Hit probability is not a distribution function (0-1).

Output: A set of targets m with their updated rewards such that target selection can be made to maximize the total Figure of Merit.

IV. **Reinforcement Learning**

Reinforcement Learning (RL) [5] is classified as a third machine learning model, with supervised learning and unsupervised learning as the basic two models. Reinforcement

learning problems involve a reward or penalty based learning approach, where an agent must explore and decide what to do - how to map situations to actions, so that at the end of the game, he maximises a numerical reward. What makes RL different from other machine-learning models is the fact that the agent is not trained which actions to take, but instead must explore and discover by trial and error to identify the actions yielding the highest reward.

Reinforcement Learning (RL) is inspired by behavioural psychology and operates by learning optimal policies through trial and error. Unlike supervised learning, RL does not rely on labelled data; instead, it receives feedback in the form of rewards. A typical, RL problem-solving loop [9] is depicted in Figure 1. The agent gathers the information from the environment which is considered as a sensory stimulus in choosing the next action. Based upon the current state of the environment, the agent chooses an action and upon execution of it receives a reward. Depending upon the appropriateness of the action, the reward can be positive or negative. A positive reward adds more weightage to the selected action, whereas a negative reward on an action is considered as a bad outcome in terms of globally set goals. A reward history of every action is accumulated over a number of trails and is used as a record book for selecting the next action. Thus, based on the output of the action taken, the environment might change, thereby affecting the states and actions possible in the future

A. Core Components of RL

An RL problem is typically modelled as a Markov Decision Process (MDP), represented by a 5-tuple (S, A, P, R, γ):

- S: Set of states
- A: Set of actions
- P: Transition probability function $P(s'|s, a)$
- R: Reward function $R(s, a)$
- γ: Discount factor, $0 \leq \gamma < 1$

The agents objective is to learn a policy $\pi(a|s)$ that maximizes the expected cumulative reward.

B. The Reinforcement Learning Process

At each time step t:

1) The agent observes state s_t
2) Chooses action a_t according to policy π
3) Receives reward r_t and observes next state s_{t+1}
4) Updates its knowledge to improve future decisions

C. Q-Learning Algorithm

Q-Learning [6] is a model-free, off-policy RL algorithm that learns the action-value function $Q(s, a)$:

1) Q-Value Update Rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right]$$

- α: Learning rate
- γ: Discount factor
- r_t: Immediate reward
- $Q(s, a)$: Estimated future value of taking action a in state s

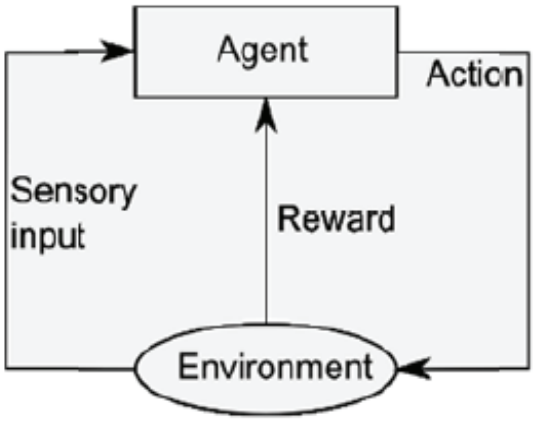

Fig. 1: *RL Loop*

2) Algorithm Steps:

1) Initialize $Q(s, a)$ arbitrarily
2) For each episode:
   a) Initialize s
   b) Repeat for each step of the episode:
      i) Choose a from s using an ϵ-greedy policy
      ii) Take action a, observe r and s'
      iii) Update $Q(s, a)$ using the update rule
      iv) $s \leftarrow s'$

D. Exploration vs Exploitation

A distinctive feature of RL is finding a balance between exploration (attempting new actions) and exploitation (taking advantage of known information). This is commonly managed via ϵ-greedy strategies.

1) Merits of RL based approach: The fundamental merit of RL lies in its quick adaptation to a changing environment. Every action is quantified in terms of its reward, indicating the measure of appropriateness of the actions taken. Having the feedback of the action in loop, the agent then adjusts its memory to select a better action next time. Therefore, RL algorithms discovers a policy that maps states of the environment to the actions the agent should take in those states. To summarize, based on its current state, the agent decides best action. Another distinctive feature of reinforcement learning is that the agent deals with an uncertain environment, i.e., the reward of an action is a not always certain. This is in contrast with many approaches that consider sub problems without addressing how they might fit into a larger picture.

Though the information or knowledge of the environment is not known completely, the agent has to decide his action policy. This kind of learning based action selection strategy is important and useful in applications such as Target Selection where the basic operational strategy is clear, but a critical decision to be made considering the uncertainty of target engagement in the heat of battle.

2) Challenges in RL: The most challenging task in reinforcement learning when compared to other kinds of learning is the trade-off between exploration and exploitation. During

exploitation, a reinforcement learning agent prefers actions that it has tried in the past and found to be effective in producing high reward. But to identify an action which gives good rewards, the agent must explore. Therefore, neither exclusive exploration nor exploitation works best. Exploitation results in short term gains, whereas exploration leads to long term gains. Thus, the agent must learn and update by choosing a variety of actions and gradually favour those that appear to be best. However, the rate of exploration and exploitation depends on the distribution of the uncertainty factor of the rewards.

V. Solution with Multi Armed Bandit (MAB) based RL Approach

In this segment, we discuss upon *multi-armed bandits*, a class of reinforcement learning problems where the problem solution is framed by selecting a sequence of actions while observing the sequence of rewards. One of the distinctive feature of multi-armed bandits is that the environment is *not* modified/changed by the agent based on the action list. The objective is to discover the best actions by exploration, and exploit those offering the highest rewards.

Multi-armed bandit (MAB) problem [13], is regarded as a class of sequential resource allocation problem which seeks a solution for an optimal allocation of resources among the available set of competing options. There exists a fundamental conflict in decision making a conflict between making decisions that yield high current rewards (*short term gains*), versus making decisions that sacrifice current gains with the prospect of better future rewards (*long term gains*).

In k-armed bandit problem, there are k actions, and each action has an expected or mean reward denoted as *value* of that action. The action selected on time step *t* is denoted as *At*, with the associated reward at time step *t* as *Rt*. The *value* then of an arbitrary action *a*, denoted by *q*(a)*, is the expected reward given that *a* is selected:

$$q^*(a) := E[Rt|At = a]$$

It is simple to solve the k-armed bandit problem by choosing an action with highest outcome, provided if the value of each action is deterministic. In situations where action values are not certain, only estimates may be made available. The estimated value of action *a* at time step *t* is denoted as *Qt(a)*. It is expected that *Qt(a)* to be close to *q*(a)*.

Based on the estimates made for the actions upto time step *t*, it is possible to select an action whose estimated value is the greatest. Such an action is called as ***greedy actions***. When one of greedy action is selected, it indicates that we are exploiting the knowledge base of the estimated values of the actions. If instead one of the non-greedy actions is selected, then it is meant to be ***Exploring***, because this enables our improvement in estimation of non-greedy actions value. As said earlier, exploitation is the right thing to do to maximize the current reward, but exploration is required to maximise total reward in the long run. Whether it is better to explore or exploit is a complex criteria which depends on the nature of the problem, precise values of the estimates, uncertainties, and the number of iterations considered in finding a solution.

Essentially, MAB algorithm comprises of two major steps to arrive at the optimal action list i) Estimation of the reward of the selected action ii) Action selection policy. Action values are estimated using sample-average method. The outcome of every action is stored and averaged out over a number of samples. Naturally, by the virtue of the law of large numbers, with the increase in the number of iterations, estimates converges to actual values i.e *Qt(a)* converges to *q*(a)*. Having done with the estimates, next question now to address is, the criteria to be adopted to select an action. One obvious solution is to select the action having the highest estimated value as

$$At := argmax(Qt(a))$$

If an action having the highest reward so far is selected all the time then it is termed as Greedy action selection policy, which always exploits current knowledge to maximize immediate reward. It spends no time at all to sample apparently inferior actions to check if they produce really better rewards. A straightforward substitute to this policy is to act greedily most of the time, but once in a while, with small probability *epsilon*, select one of the non-greedy action, independent of the action-value estimates. These methods are coined as *epsilon* $-$greedy methods. Using this approach, new actions will be explored and with the increase in number of iterations every action will be sampled an infinite number of times, thus ensuring that all the *Qt(a)* converge to *q*(a)* .

Here an analogy is made between k-armed bandit problem and the optimal Target selection problem.

- k in k-armed bandit problem represents the number of Targets.
- Reward of selecting a Target is analogous to the Figure of Merit of the Target.
- Hit Probability figures are similar to the random nature of each action reward.

Now, it is required to update the rewards list of the Targets, using which an optimal selection of Targets can be made so that the total Figure of Merit is maximised.

A. Implementation Methodology & Performance Remarks

As explained earlier, reward estimation is done using sample average method. Let *Ri* represent the received reward of an action *a* after the ith selection, and let *Qn* signify the estimate of its action value after it has been selected *n - 1* times, which we can be stated as

$$Qn := \frac{R1 + R2 + ... + Rn - 1}{n - 1}$$

To minimise the computational costs, the above equation can be framed as an incremental formula as follows

$$Qn + 1 := \frac{Qn + [Rn - Qn]}{n}$$

VI. Case Study, Experimental Results and Concluding Remarks with MAB

For illustration purpose, a case study with 10 number of Targets i.e, *n = 10* is considered. Each Target is given a Figure of Merit *FMn* which is nothing but the reward gained the attacker, if a particular Target *Tn* is hit. In addition each

Target Number (T_n)	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
Figure of Merit (FM_n)	10	9	8	7	7	7	6	6	5	4
Kill Probability of the Target (P_n)	0.7	0.8	0.9	0.65	0.75	0.85	0.8	0.8	0.99	0.9

Fig. 2: *Target Details*

Target is also associated with a hit probability number *Pn* which indicates the kill probability of the Target *Tn*. The details of the case study under analysis is shown in Figure 2.

A. Case Study

The case study is carefully crafted to depict various possible scenarios which are detailed as follows.

- Consider the Targets list *T1, T2, T3*, among the three Targets, Figure of Merit of *T1* is the highest, but the kill probability of *T1* is the lowest. Under such scenario selecting the Target, *T1* may not become an optimum choice.
- Consider the Targets list *T4, T5, T6*, among the three Targets, Figure of Merit of all the Targets is same, but the kill probability of *T4* is the lowest. Under such situations, Target, *T6* should become an optimum choice.
- Consider the Targets list *T7, T8*, Figure of Merit of *T1* and kill probability of both the Targets are same. Under such conditions, both the Targets should be given equal importance.
- Consider the Targets list *T9, T10*, among the two, Figure of Merit of *T9* and the kill probability of *T9* is the highest. Given the situation selecting the Target, *T9* should be an optimum choice.

As it can be seen from the Target data, had there been no uncertainty element involved in the problem, it could be solved with some difficulty. Unfortunately, the stochastic nature of the rewards makes its hard to craft the solution using pen and paper. Moreover, the increase in the variation of rewards and uncertainties, one cannot find a solution informally. The experimental results sub-section demonstrates how this can be attempted using variants of k-armed bandit solution strategies.

B. Experimental Results using MAB

Here, in this section the results of bandit algorithm are evaluated for the case study shown in the previous subsection. Two major aspects of MAB are considered during the process of evaluation. Primarily, it is studied whether the bandit algorithms have a potential in arriving at an optimal solution or not. To make evident of this aspect, different practical case studies are taken up. Secondly, the aspect of exploration and exploitation trade-off for the given problem is deliberated.

To demonstrate the potential of bandit algorithm in arriving at an optimal target list, we have considered four practical situations having various possible rewards and probability structures. Every indicative scenario is subjected to MAB algorithm and results of which are shown below:

- Case1: This case pertains to the original rewards and probability figures of the case study under consideration. Intuitively, for the given rewards and the associated probabilities, the optimal target selection in the order of preference would be *T2/T3, T1,T6,T5/T9, T4/T8, T7,T10*. However, the updated target rewards as shown in Figure 3 after running with epsilon greedy algorithm *does not follow the intuition* which clearly point out that such problems cannot be addressed by intuition and informal methods.
- Case2: Here, in this case we consider a situation where there is no uncertainty in the estimates made. Thus, the probability of the estimated reward is always 1. Hence, MAB should always select an action which is having the highest Figure of Merit i.e. *T1* should be chosen. The results shown in Figure 4 confirms the same.
- Case3: This case depicts a condition, where every action provides an equal reward, however, the probability of getting the reward is not certain. Logically, as the outcome of every action is equal, the one which has the lowest uncertainty factor to be given importance during the action selection policy. The result shown in Figure 5 demonstrates the same.
- Case4: This case corresponds to an impractical scenario, where the problem is much simplified with all the actions having the same outcome and with no element of uncertainty in the estimates made. In this ideal case, there is no competition between the actions and every action should be given equal weightage. The results of Figure 6 confirms the analogy.

C. Drawbacks using epsilon greedy MAB

Basically, there are two problems associated with epsilon greedy MAB problem solution strategy detailed as follows

- Firstly, there is a trade-off between exploitation and exploration using MAB. As discussed earlier, during exploration non-greedy actions are selected to assess the reward distribution of every other action, whereas, while during exploitation, the action having the better estimated reward is considered. If we go with pure exploration, as the acquired knowledge is not leveraged,

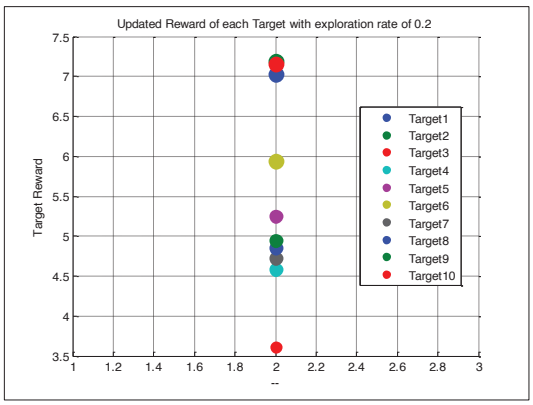

Fig. 3: *Updated Target Rewards using MAB for Case-1*

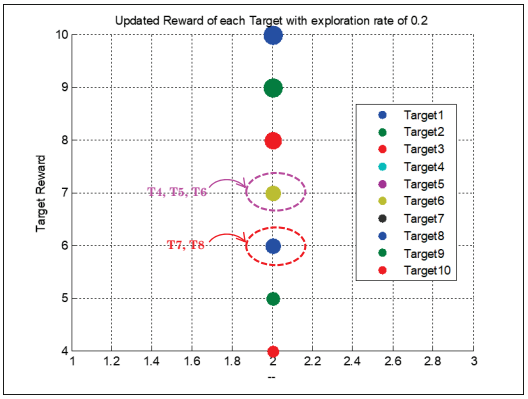

Fig. 4: *Updated Target Rewards using MAB for Case-2*

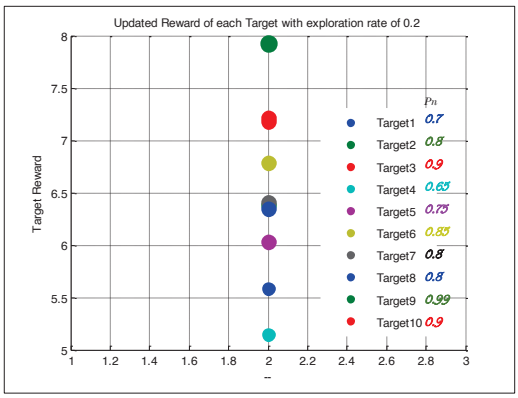

Fig. 5: *Updated Target Rewards using MAB for Case-3*

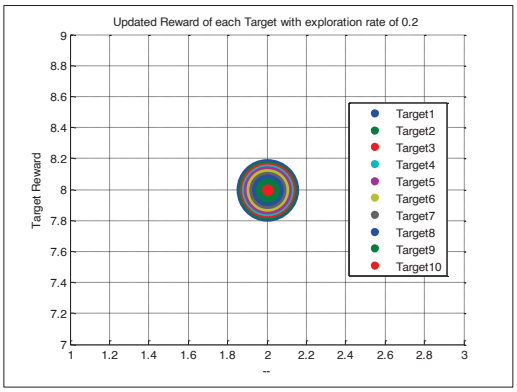

Fig. 6: *Updated Target Rewards using MAB for Case-4*

it may result may lead to sub-optimal rewards. On contrary, by following a pure exploitative strategy we might miss out on discovering potentially more rewarding actions. Hence, both exploration and exploitation is required, but the question comes here is what should be the exploration rate. As stated earlier, the exploration rate purely depends on the distribution of the certainty factor, rewards of action of the data set under consideration. The results illustrated in previous section corresponds to an exploration rate of 0.2 i.e., a new action is explored in 20 percent of the trails, whereas in 80 percent of the trails, an action is selected having the maximum reward so far. The exploration rate can be decided only using hit and trial basis.

- Secondly, during exploration phase, selection of non-greedy actions is random in nature. Every action is selected with equal probability. The underlying problem with this action selection policy is that, after running the algorithm for sufficiently large number of iterations, even if an action is found to produce less reward, then also it is given equal weightage. To summarize, during exploration phase, due to the random nature of action selection policy, less rewarded action is also given equal imporatnce. Therefore, action selection policy to be modified such that potentially high reward producing actions to be explored in order to get an optimal estimated value.

The above said problems are demonstrated in Figure 7. As it can be seen from the data provided in the table, Target priority order with various exploration rates is different. However, the Target priority order with exploration rate of 0.2 and 0.5 is a bit comparable. Deviation in the priority order in these two cases is attributed to the random phenomenon of selection the action for exploration. There is no weightage given to the uncertainty factor in choosing an action during exploration phase. These two issues are typically addressed by Upper Confidence Bound based solution strategy. The details of which are explained in the subsequent section.

Target Rewards with various exploration rates

ε / Fm	0.05	0.1	0.2	0.5
T1	6.93	7.13	6.85	6.81
T2	6.38	7.15	7.25	7.19
T3	7.25	6.83	7.19	7.21
T4	4.34	5.12	4.16	4.28
T5	5.67	5.53	5.38	5.07
T6	4.52	6.19	5.69	5.88
T7	4.00	4.17	4.11	4.83
T8	5.14	5.41	4.66	5.07
T9	5.00	5.00	4.91	5.00
T10	3.65	3.17	3.36	3.66

Target Priority order with various exploration rates

ε = 0.05	ε = 0.1	ε = 0.2	ε = 0.5
T3	T2	T2	T3
T1	T1	T3	T2
T2	T3	T1	T1
T5	T6	T6	T6
T8	T5	T5	T5
T9	T8	T9	T8
T6	T4	T8	T9
T4	T9	T4	T4
T7	T7	T7	T7
T10	T10	T10	T10

Fig. 7: *Target Rewards and Priority Order for various exploration rates*

VII. Improved Algorithm - Upper Confidence Bound (UCB) based solution

A. Introduction to UCB

In feedback based learning mechanism like RL, both exploration and exploitation are required as the estimates of the action values are uncertain. Exploration, essentially tries to estimate the outcome of non-greedy actions, which may found to be better than the greedy actions. With epsilon-greedy method, during exploration phase, non-greedy actions are tried, however, the action selection policy adopted during this phase indiscriminately selects every action with the

same preference. It is ideal to explore those non-greedy actions which sound promising in producing greater rewards. It would be better to select among the non-greedy actions according to their potential for actually being optimal, taking into account both how close their estimates are to being maximal and the uncertainties in those estimates. The same principle is adopted in UCB based strategy. In UCB algorithm, a confidence interval-based approach is followed which balances the trade-off between exploration-exploitation. Unlike MAB, in UCB strategy, there are no two different phases like - exploration and exploitation. In fact, a unified strategy is followed by a revised action selection policy. The underlying principle behind the action selection policy in UCB is that - the actions are selected based not only on their estimated rewards but also on the on the uncertainty or confidence in these estimates. The UCB algorithm comprises of following 3 phase:

- Initialization Phase: Estimated reward and count of each action to be initialised.
- Action Selection Phase: In every iteration, the arm having the highest UCB value to be selected. Here, the UCB value is a combination of the estimated reward and a term that represents the uncertainty or confidence interval.
- Update Phase: Followed by action selection phase, update the number of counts and the estimated reward for an action.

Mathematically, the UCB value for an action a at time t is given by:

$$At = argmax[Qt(a) + c * \sqrt{\frac{ln(t)}{Nt(a)}}]$$

where the term *Qt(a)* represents the estimated reward, the number $c > 0$ controls the degree of exploration, the sqaure-root term represents the confidence interval, *ln t* denotes the natural logarithm of t and *Nt(a)* represents number of times a particular action a is selected.

In the above equation, the estimated reward term *Qt(a)* encourages exploitation by favouring actions with higher known rewards, whereas, the confidence interval term encourages exploration by favoring actions that have been selected fewer times, thus accounting for the uncertainty in their reward estimates.

The square-root term used in the equation of action selection policy of UCB is a direct measure of the uncertainty or variance in the estimate of *a's* value. The quantity being maximised over is thus a sort of upper bound on the possible true value of action *a*, with the *c* parameter determining the confidence level. Whenever an action *a* is selected, *Nt(a)* is incremented, the uncertainty associated with that action is apparently reduced. On the other hand, each time an action other than *a* is selected *t* is increased, the uncertainty estimate is increased. The amount of increase in the uncertainty estimate gets smaller over time because of the use of natural logarithm term *ln(t)*.As a result, eventually all actions will be selected. However, actions with a lower estimated value or the actions that have been selected more number of times will be selected with a lower frequency. In this way, UCB algorithm maintains a balance between exploration and exploitation through its action selection policy.

B. Experimental results with UCB

The same example case study which is subjected to solution strategy using MAB is subjected to UCB algorithm. Experimental results which details the updated Target rewards with uncertainty in place using UCB are demonstrated in Figure 8 . A comparison of rewards and Target priority order

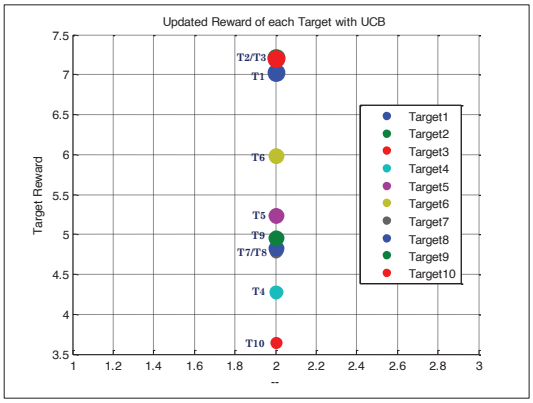

Fig. 8: *Target Rewards with UCB*

with MAB (*epsilon* of 0.2) and UCB is shown in Figure 9.

Target Rewards with MAB and UCB

	MAB (ε = 0.2)	UCB
T1	6.93	7.0
T2	6.38	7.21
T3	7.25	7.19
T4	4.34	4.28
T5	5.67	5.24
T6	4.52	5.98
T7	4.00	4.78
T8	5.14	4.83
T9	5.00	4.96
T10	3.65	3.64

Target Priority order with MAB and UCB

MAB (ε = 0.2)	UCB
T2	T2
T3	T3
T1	T1
T6	T6
T5	T5
T9	T9
T8	T7
T4	T8
T7	T4
T10	T10

Fig. 9: *Comparison of Rewards and Target Priority Order - MAB vs UCB*

From the comparison data, it is evident that Target rewards and the Target priority order with UCB is in close match with the MAB results. However, there is a difference in priority order of Targets *T4 vs T7/T8* . Taking the reward value and probability of hit into consideration for *T4 vs T7/T8*, T7/T8 is the better choice. This clearly indicates the drawback in action selection policy of MAB algorithm during exploration phase. Also, as with MAB, in UCB there is no uncertainty in the updated rewards of the actions as there is no randomness involved in selecting an arm. Thus, the results provided with

this updated strategy are consistent, proving the efficacy of UCB over traditional MAB based solution strategy.

VIII. Summary and Conclusion

In this paper, the comparison of performance of two different methods of Reinforcement Learning (RL) algorithm (i) Solution with Multi Armed Bandit (MAB) based approach and (ii) Upper Confidence Bound (UCB) based approach is discussed for a typical optimal Target Selection strategy taking into consideration of the uncertainties associated with the hit probability of a Target in order to maximise the objectives of a mission. Firstly, the problem was showcased as an equivalent to multi-armed bandit problem which is the simplest form of RL. Multi-armed bandit (MAB) solutions employ variants of simple greedy algorithms to find an optimal long term reward. It is seen that inherent issues with MAB are balancing between exploitation and exploration phase and action selection policy. These limitations of MAB are addressed by considering Upper Confidence Based (UCB) solution strategy. It is seen that in UCB based approach, there is no uncertainty in the updated rewards of the actions, as there is no randomness involved in selecting an arm. The results provided with this updated strategy are consistent, proving the efficacy of UCB over traditional MAB based solution strategy.

References

[1] P. P. Bhattacharjee and D. Ghosh, "A survey on the weapon target assignment problem", *Journal of Defense Modeling and Simulation*, vol. 15, no. 1, pp. 524, 2018.

[2] A. Washburn, "Two-person zero-sum games for weapon allocation", *Operations Research*, vol. 30, no. 2, pp. 256275, 1982.

[3] R H Day, "Allocating weapons to target complexes by means of nonlinear programming", *Operations Research*, vol. 14, pp.992-1013, 1966.

[4] A.S. Manne, "A target-assignment problem", *Operational reaserach*, vol. 6, pp.346–351, 1958.

[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed., MIT Press, 2018.

[6] C. J. C. H. Watkins and P. Dayan, "Q-learning", *Machine Learning*, vol. 8, pp. 279292, 1992.

[7] B.Gaudet, K.Drozd, and R.Furfaro,"Deep reinforcement learning for weapons to targets assignment in a hypersonic strike", vol.10, 2023.

[8] O.S. Heon, G.W. Byueon, Y.I. Cho, S.Kwon, and J.H. Woo, "Artificial intelligence in combat decision-making: weapon target assignment via reinforcement learning and graph neural networks", *Authorea Preprints*, 2024.

[9] H.R. Hildegarde Mouton, Jan Roodt ,"Applying reinforcement learning to the weapon assignment problem in air defence", *Journal of Military Studies, South Africa*, 39:99–116, 2011.

[10] J.Li, G.Wu, and L.Wang, " A comprehensive survey of weapon target assignment problem: Model, algorithm, and application", *Engineering Applications of Artificial Intelligence*, 137:109212, 2024.

[11] S.Li, X.He, X.Xu, T.Zhao, C.Song, and J.Li, " Weapon-target assignment strategy in joint combat decision-making based on multi-head deep reinforcement learning", *IEEE Access*, 11:113740–113751, 2023.

[12] J.M. Rosenberger, H.S. Hwang, R.P. Pallerla, A.Yucel, R.L. Wilson, and E.G. Brungardt, " The generalized weapon target assignment problem", *10th International Command and Control Research and Technology Symposium*, pages 1–12. Citeseer, 2005.

[13] C.S. Tor Lattimore, "Bandit algorithms", Online content.

[14] T.Wang, L.Fu, Z.Wei, Y.Zhou, and S.Gao, "Unmanned ground weapon target assignment based on deep q-learning network with an improved multi-objective artificial bee colony algorithm", *Engineering Applications of Artificial Intelligence*,117:105612, 2023.