Journal of Korean Society of Industrial and Systems Engineering
https://doi.org/10.11627/jksie.2022.45.4.042

Vol. 45, No. 4 : 42-52, December 2022
ISSN : 2005-0461 / EISSN : 2287-7975

Reinforcement Learning-based Dynamic Weapon Assignment to Multi-Caliber Long-Range Artillery Attacks

Hyeonho Kim* · Jung Hun Kim · Joohoe Kong*** · Ji Hoon Kyung*†**

*Department of Industrial Engineering, Hannam University
**Hanwha Systems
***CSB LAB

다종 장사정포 공격에 대한 강화학습 기반의 동적 무기할당

김현호* · 김정훈** · 공주회*** · 경지훈*†

*한남대학교 산업공학과
**한화 시스템
***씨에스비랩

North Korea continues to upgrade and display its long-range rocket launchers to emphasize its military strength. Recently Republic of Korea kicked off the development of anti-artillery interception system similar to Israel's "Iron Dome", designed to protect against North Korea's arsenal of long-range rockets. The system may not work smoothly without the function assigning interceptors to incoming various-caliber artillery rockets. We view the assignment task as a dynamic weapon target assignment (DWTA) problem. DWTA is a multistage decision process in which decision in a stage affects decision processes and its results in the subsequent stages. We represent the DWTA problem as a Markov decision process (MDP). Distance from Seoul to North Korea's multiple rocket launchers positioned near the border, limits the processing time of the model solver within only a few second. It is impossible to compute the exact optimal solution within the allowed time interval due to the curse of dimensionality inherently in MDP model of practical DWTA problem. We apply two reinforcement-based algorithms to get the approximate solution of the MDP model within the time limit. To check the quality of the approximate solution, we adopt Shoot-Shoot-Look(SSL) policy as a baseline. Simulation results showed that both algorithms provide better solution than the solution from the baseline strategy.

Keywords : Reinforcement Learning, Dynamic Weapon Target Assignment, Long-Range Artillery

1. 서 론

북한은 탄도탄과 더불어 북방한계선(MDL) 인근에 장사정포를 배치하여 대한민국의 수도권을 위협하고 있으며, 최근 다종 다양한 탄도탄을 혼합하여 발사하는 형태를 보임에 따라 장사정포 공격에도 적용될 것으로 예상된다. 북한은 개전 초 수 백여 문의 장사정포를 이용하여 수도권의 군사시설 및 국가 중요시설을 무력화하기 위하여 일제히 공격할 것으로 판단된다[18]. 장사정포 공격 대응방안으로는 발사체를 요격하는 방식과 발사 플랫폼에 대해 직접 타격하는 방식으로 구분할 수 있다. 전자의 경우 사드나 패트리어트로 요격이 가능하나, 비용 대 효과상 비효율적인 대응 방안으로 고려대상이 아니다.

Received 15 September 2022; Finally Revised 7 November 2022; Accepted 9 November 2022
† Corresponding Author : kjh@hnu.kr

현재까지 장사정포의 공격에 대응하는 방식은 장사정포 진지를 직접 타격하여 사전에 무력화 시키거나 아군이 공격을 받으면 가용 무기체계를 이용하여 대응사격을 하는 방식이다[14]. 그러나, 이러한 대응방식은 개전 초에 북한의 갱도화 된 포병진지를 선제타격 하더라도 무력화를 보장할 수 없기 때문에, 이스라엘의 아이언 돔과 유사한 한국형 장사정포 요격체계 개발을 추진하고 있다[25]. 한국형 장사정포 요격체계는 무력화 되지 않은 적 장사정포병이 수도권에 대하여 일제히 사격을 가할 때 돔 형태의 방어막을 형성하여 아군의 피해를 상당수 줄이는 개념이다[6].

이러한 한국형 장사정포 요격체계 개발을 위해서는 하드웨어와 더불어 최단시간 내에 효과적으로 사격할 수 있도록 표적을 할당해 주는 무기표적할당 알고리즘 등 소프트웨어 개발이 필수적이다. 특히, 한반도 전장 환경 특성상, 적 장사정포 진지와 수도권은 매우 근거리이기 때문에, 최단시간 내 대응할 수 있는 무기체계 교전알고리즘이 매우 중요하다.

기존의 장사정포 대응방법은 요격 유도탄 운용자의 판단에 의해서 대응하는 방법으로 대응 시간의 차이가 발생하고, 피폭 받는 아군측 자산의 가치, 아군측의 포탄 재고, 적군의 추정 포탄 재고 등 전장의 상황에 따라 공격의 대응에 영향을 주는 제반 요소의 변화를 반영하지 않고 고정된 대응방법을 가진다는 제한점이 상존한다.

따라서, 한국형 장사정포 요격체계를 개발할 때, 요격이 가능한 시간 안에 실시간 전장 상황을 반영하여 우수한 교전 효과를 도출할 수 있는 무기 표적 할당의 수립이 가능한지를 살펴보는 것은 매우 중요하다. 현재까지 연구된 휴리스틱 알고리즘기반의 무장할당 방법은 주어진 시간 내에 정확한 최적해를 산출하는 것이 불가능하여 강화학습 기반의 개략 최적해를 구하는 무장 할당 알고리즘 적용이 필수적이다.

본 논문에서는 기존 장사정포와 성능이 한층 개량된 장사정포를 혼합하여 발사하는 전술을 사용하는 경우, 강화학습 기반의 알고리즘들을 적용하였을 때, 무기 표적 할당의 결과가 우수한 지를 시뮬레이션을 실행하여 살펴보고, 알고리즘의 실행 시간을 기록하여 실제 상황에서 사용 가능한 지 알아본다. 본 논문의 구성은 다음과 같다. 제2장에서는 관련 연구를 요약하고, 제3장에서는 문제 정의와 방법론을 설명한다. 제4장에서는 시뮬레이션 내용 및 결과를 요약하고 분석하며, 제5장에서 결론을 맺는다.

2. 관련연구

무기 표적 할당(WTA, Weapon Target Assignment) 문제는 다수의 표적에 가용 무기 자원을 적절히 할당함으로써 보호하고자 하는 자산의 생존 가치를 높이거나, 표적의 생존 가치를 낮추는 문제이다. 무기 표적 할당 문제는 크게 정적(static) 무기표적할당 문제와 동적(dynamic) 무기 표적 할당 문제로 구분한다[9]. 정적 무기 할당 문제에서는 한번에 모든 무기를 목표에 할당하는 반면, 동적 무기 표적 할당 문제에서는 연속적인 여러 단계에 걸쳐 의사결정을 실행한다.

무기 표적 할당 문제에서는, 최적해를 찾기 위한 알고리즘에 관한 연구가 무기와 표적의 수가 증가함에 따라 계산시간이 오래 걸리기 때문에, 최근에는 신속한 의사결정을 위해, 모형을 선형화 하거나[11], 유전자 알고리즘[13], 개미군집 알고리즘[16] 등과 같은 휴리스틱 알고리즘을 활용한 근사 해를 찾는 연구가 활발히 진행되고 있다.

Naeem et al.[19]은 동적 무기 표적 할당문제를 실시간 스케줄링 문제의 특별한 사례로 보고, 특정 제약조건하에서 무기와 표적을 각각 기계와 처리되어야 하는 작업으로 가정하여 접근하였다. 이때 안정된 결혼 알고리즘(Stable Marriage Algorithm)을 변형하여 표적의 위협을 계산하고 무기-표적을 할당하는 2단계 접근방식을 사용하였다. Cha et al.[4]은 포병 부대의 표적 할당 및 사격일정계획 수립을 위하여 동적계획법으로 표적-포병부대를 할당하고 분기한정 알고리즘을 사용하여 포병부대의 사격 스케줄을 결정하는 2단계 휴리스틱 알고리즘을 제안하였다. 하지만 불확실한 교전상황이 존재하는 동적 무기 표적 할당에 스케줄링 알고리즘을 적용한 연구는 문제 특성과 시간효율성을 충족시키기 위해 제약조건을 단순화하는 등 조건에 따라 모든 사례에 적용하기 어렵다는 한계점이 있다[12]. Jung et al.[12]은 동적으로 무기를 할당하고, 교전계획을 작성하는 휴리스틱 기반의 Rolling-Horizon 스케줄링 알고리즘을 제안하였으며, 기존의 SLS 전략보다 우수한 해를 도출할 수 있음을 보여 주었다.

Bertsekas et al.[1]은 강화학습 기반의 알고리즘을 동적 무기 표적 할당 문제에 적용한 최초의 논문으로, 미사일 방어 문제에서 현재의 표적에 몇 발의 무기를 배정하고, 이후 공격에 대비하기 위하여 얼마의 재고를 남겨야 하는 문제를 다루었다. 논문에서 제시한 neuro-dynamic programming은 최적 함수를 근사화 하기 위하여 인공신경망과 시뮬레이션을 활용 하였다. Davis et al.[7]은 감시 위성의 발달에 따라 공격하고자 하는 자산의 파괴 정도를 공격측에서도 실시간으로 파악하고 있다고 가정 하였다. 아울러 공격 측은 공격 하고자 하는 자산의 실시간 가치를 반영하여 확률적으로 공격하며, 방어 측에서도 방어 자산의 실시간 가치를 파악하고 있다고 가정하여 현대전의 기술발달을 반영하였다. 저자들은 4가지 미사일 방어 시나리오에 대한 동적 무기 표적 할당 문제를 ADP(Approximate

Dynamic Programming) 알고리즘을 적용한 시뮬레이션 결과가 최적해에 비하여 8%~22% 차이를 보였음을 보고 하였다. Summers et al.[24]은 교전 기간, 요격 유도탄의 종류 등을 다양화한 시나리오에 대하여 공격 미사일 방어를 위한 동적 무기 표적 할당 문제를 ADP 알고리즘을 적용한 해가 테스트한 모든 시나리오에서 운용자에 의한 SSL (Shoot-Shoot-Look) 발사방법보다 우수함을 제시하였다.

대부분의 동적 무기 할당 문제에 관한 연구는 미사일 방어 문제를 다루고 있다. 미사일 방어는 한 번의 공격에 수 발의 미사일 발사를 가정한다. 이와는 달리 한 번의 공격에 수십 발의 포탄을 발사할 수 있는 장사정포 방어체계에 대한 연구는 초기 단계이다[15]. 이와 관련된 연구로, 탄도탄에 대한 비행궤적의 정확한 예측이 요격 유도탄 할당에 가장 중요한 요소이므로[5, 8, 26], Im et al.[10]은 이를 바탕으로 장사정포 발사체의 궤적의 분류에 따른 규칙기반의 할당 방법론을 제안하였다. Shin et al.[23]은 교전 상황 중 실시간 무기표적 할당을 위한 멀티 에이전트 강화학습 모델을 제시하였다. 제시된 모델은 표적과 발사대 수가 증가함에 따라 학습 차원이 증가하는 문제를 평균 필드 게임 이론을 이용하여 완화시켰으며, 기존의 할당 기법과 달리 표적 수에 관계없이 일정 시간으로 할당함을 확인하였다. 이 논문의 테스트 시나리오에서 장사정포에 대한 언급은 없으나, 표적 수를 증가시키면 장사정포 공격 시나리오와 유사할 것으로 보인다. Lee et al.[15]은 장사정포 공격에 대한 방어를 언급한 최초의 동적 무기 할당 연구로, 요격 탄도탄의 재고 수준에 따른 3가지 시나리오를 시뮬레이션을 실행한 결과, 강화학습 기반의 알고리즘을 적용한 해가 SSL 발사방법보다 4%~11% 우수함을 보여 주었다.

3. 방법론

3.1 문제정의

본 논문에서 아군은 가치가 다른 여러 자산을 방어하기 위하여 다 수의 요격 포대를 배치하고 있으며, 한 포대는 여러 자산을 방어할 수 있다. 자산이 피폭을 받으면 피폭한 포탄의 발수에 비례하여 가치가 감소하며, 자산의 실시간 가치는 적군과 아군 모두 실시간으로 파악하고 있다. 요격 포대는 초기에 포탄 재고를 가지고 있으며, 재고가 소진되면 더 이상 포탄의 보충은 없다. 적군은 여러 종류의 장사정포를 가지고 있다. 적군은 아군 자산의 가치에 비례하여 확률적으로 자산을 향하여 포탄을 발사한다. 적군의 포탄 재고는 파악되지 않으므로, 아군 자산의 가치가 0이 될 때까지 계속 공격하는 것으로 가정한다. 아군은 적군의 연속적인 공격으로부터 최대한 자산 가치를 방어한다.

3.2 MDP 모형

MDP(Markov Decision Process) 모형은 각 의사결정 시점에서의 상태(state), 행동(action), 전이확률(transition probability), 보상(reward), 할인인자(discount factor)의 5가지 항목에 의해 정의되며 다음과 같이 요약된다[22].

의사결정시점: 의사 결정이 내려지는 한 시점으로 시간 t로 나타낸다. $T=1,2,\cdots,T,\ T\le\infty$ 는 의사결정시점의 집합이다.

상태: 모든 가능한 시스템의 상태들의 집합은 S 로 나타내며, t시점에서의 시스템의 상태는 S_t로 나타낸다.

행동: t시점에서 시스템의 상태가 S_t일 때 의사 결정자가 결정하는 행동으로 보통 a_t로 나타낸다. 이 행동으로 시스템의 상태를 변화시켜 t+1 시점에서의 시스템의 상태가 새롭게 된다.

보상: t시점에서 시스템의 상태가 S_t일 때 의사 결정자가 결정하는 행동 a_t으로 인해 받는 보상으로 실수 값을 갖는 함수 $C(S_t, a_t)$로 표현된다.

전이확률: t시점에서 시스템의 상태가 S_t일 때 의사 결정자가 결정하는 행동 a_t으로 t시점에서 각 새로운 상태별로 변하게 되는 확률을 의미한다.

할인인자: 현재 얻게 되는 보상이 미래에 얻게 될 보상보다 얼마나 더 중요한지를 나타내는 값으로, $\gamma\in[0,1]$로 나타낸다.

이상은 일반적인 MDP 모형에 관한 설명으로, 본 논문에서 다루는 동적 무기 표적 할당 문제를 표현하기 위하여 다음의 기호들을 사용한다.

의사 결정 시점 t에서 요격 포대의 유도탄 재고는 $R_t=(R_{ti})_{(i\in A)}\equiv(R_{t1}, R_{t2},\cdots, R_{t|A|})$로 나타내며, 여기에서 $R_{ti}\in\{0,1,\cdots,r_i\}$ 이다. R_{ti}는 t에서의 A에 속하는 i에 위치한 요격 포대의 재고다.

요격 포대가 방어하는 자산의 상태는 A_t로 나타내며 $A_t=(A_{ti})_{i\in A}\equiv(A_{t1}, A_{t2}, ..., A_{t|A|})\quad A=\{1, 2, ..., |A|\}$ 는 모든 자산의 집합이고, $0\le A_{ti}\le 1$이다. A_{ti}는 의사결정 시점 t에서 A에 속한 자산 i의 상태를 나타내며, 자산의 몇 퍼센트가 장사정포로부터 피해를 입지 않고 남아있는지를 보여준다.

장사정포의 공격 벡터는 $\widehat{M}_t=(\widehat{M_{tij}})_{i\in A, j\in J}$로서 표현되며, $\widehat{M_{tij}}\subseteq\widehat{M_{tj}}$는 t에서의 자산 $i\in A$를 표적으로 삼는 $j\in J$ 형태의 장사정포 공격이다.

이러한 구성요소를 사용하여, t시점에서의 시스템의 상

태는 각 자산의 상태, 각 요격 포대의 유도탄 재고, 장사정포의 공격 벡터로 나타내며, $S_t=(A_t, R_t, \widehat{M}_t)\in S$=로 표현한다. $x_{tijk}\in N_0$는 t시점에서 $k\in\widehat{M}_{tij}$의 장사정포탄을 향하여 자산 $i\in A$를 방어하기 위하여 요격 포탄의 수라고 하자.

의사 결정 벡터 $x_t=(x_{tijk})_{i\in A, j\in J, k\in\widehat{M}_{tij}}$=는 t시점에서 장사정포 공격에 대한 요격포의 할당을 나타내는 요격 유도탄 벡터다. 그러면, 방어 측의 모든 가능한 행동들의 집합은 다음과 같다.

$$X_{S_t}=\left\{x_t : \sum_{j\in J}\sum_{k\in\widehat{M}_{tij}} x_{tijk}\le \min\{R_{ti}, x^{max}\},\ \forall i\in A\right\} \quad (1)$$

위 식에서 x^{max}는 적군의 공격을 방어하기 위하여 한 시점에서 연속해서 발사할 수 있는 유도탄의 최대치를 나타낸다.

시스템 상태의 전이를 나타내는 함수는 $S_{t+1}=S^M(S_t, x_t, W_{t+1})$로 표현하며, $W_{t+1}=(\widehat{A_{t+1}}, \widehat{M_{t+1}})$는 확률 변수다. $\widehat{A_{t+1}}$은 t에서 $\widehat{M}_t$와 x_t에 의해 결정되는 확률 변수다.

유도탄의 재고 는 아래 식과 같다.

$$R_{t+1,i}=R_{t,i}-\sum_{j\in J}\sum_{k\in\widehat{M}_{tij}} x_{tijk},\ \forall i\in A \quad (2)$$

t시점에서 즉각적인 자산 가치의 손실은 식 (3)과 같이 확률적으로 표현된다.

$$\widehat{C}(S_t, x_t, \widehat{A}_{t+1,i})=\sum_{i\in A} v_i(A_{ti}-\widehat{A}_{t+1,i}) \quad (3)$$

식 (3)에서 v_i는 자산 $i\in A$의 가치다. 비용 함수를 현재의 상태와 요격 유도탄 벡터의 함수로 표현하면 식 (4)와 같다.

$$C(S_t, x_t)=E\left[\sum_{i\in A} v_i(A_{ti}-\widehat{A}_{t+1,i})|S_t, x_t\right] \quad (4)$$

MDP 모형에서는 알고리즘을 통해 미래의 피해를 최소화할 수 있는 요격 유도탄 벡터(즉, 행동)를 결정한다. $X^\pi(S_t)$를 요격 유도탄 벡터를 결정하는 정책이라 하면 MDP의 목적함수는 다음과 같다.

$$\min_{\pi\in\Pi} E^\pi\left[E^T\left[\sum_{t=1}^{T} C(S_t, X^\pi(S_t))\right]\right] \quad (5)$$

자산이 오래 보전되면 보상이 커지도록 하기 위하여 할인인자를 사용하여 목적함수를 다시 쓰면 다음과 같다.

$$\min_{\pi\in\Pi} E^\pi\left[\sum_{t=1}^{\infty}\gamma^{t-1} C(S_t, X^\pi(S_t))\right] \quad (6)$$

t시점에서의 상태의 가치를 $J(S_t)$라 하면 최적의 정책은 Bellman 방정식이라 부르는 다음과 같은 식으로 표현된다.

$$J(S_t)=\min_{x_t\in X_t}(C(S_t, x_t)+\gamma E[J(S_{t+1})\mid S_t, x_t]) \quad (7)$$

이 Bellman 방정식의 해는 다음과 같다.

$$X^\pi(S_t)=\arg\min_{x_{t\in X_t}}(C(S_t, x_t)+\gamma E[J(S_{t+1})\mid S_t, x_t]) \quad (8)$$

3.3 강화학습 기반의 근사모형

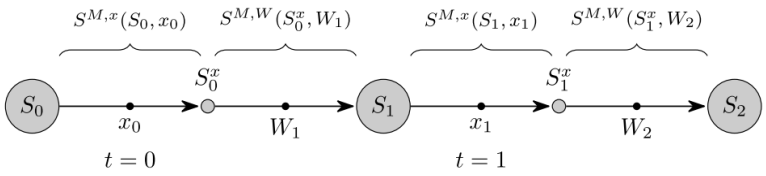
DWTA(Dynamic Weapon Target Assignment) 문제를 MDP 모형으로 표현한 후에 Bellman 방정식을 이용하여 정확한 최적해를 구할 수 있다. 그러나, 이러한 방법을 실제 문제에 적용하여 최적해를 구할 때는 두 가지 문제가 발생하며, 각 문제점과 그 해결방안을 살펴보자[20].

첫째, MDP로 모형화 하면 소규모의 문제에서는 계산이 가능하나, 문제의 크기가 커지면 계산이 불가능 하거나 너무 오랜 시간이 걸린다는 사실에 있다[20]. 예를 들어 한번 공격할 때마다 적의 장사정포가 1번에 12발 단위로 동시에 최대 3번 발사할 수 있다고 가정하자. 12발씩 10번 발사가 가능한 유도탄 재고를 가진 요격 포대가 4곳 있으며, 방어할 자산이 4곳이며, 장사정포 1발이 자산에 피격되면 자산의 최초 가치를 0.025배씩 감소시킨다고 가정하자. 이 경우 $|S_t|$는 최대 1.4×10^{12}가 된다. 여기에 $|T|$를 곱해야 전체 시스템 상태의 크기 $|S|$가 나온다. 이 사례는 DWTA 문제의 크기가 커지면 정확한 해를 구하는 방법은 사용하기 어렵다는 사실을 보여준다.

Powell[20]은 시스템 상태의 수를 줄이는 방법으로 의사결정 직후(post-decision) 상태라는 개념을 제안했다. DWTA 문제에서 의사결정 직후 상태는 요격 벡터를 결정한 직후이면서 새로운 정보 W_{t+1}는 도착하기 직전의 상태를 나타낸다. 의사결정 직후 상태를 S_t^x로 나타내면, 시스템 상태의 전이를 나타내는 함수 $S_{t+1}=S^M(S_t, x_t, W_{t+1})$는 다음과 같이 나누어 표현 할 수 있다.

$$S_t^x=S^{M,x}(S_t, x_t) \quad (9)$$

$$S_{t+1}=S^{M,W}(S_t^x, W_{t+1}) \quad (10)$$



<Figure 1> Example of a Sequence of Decisions as a MDP with Post-decision State Variable (from Fig. 2 in [22])

<Figure 1>은 $t=0$일 때와 $t=1$일 때, 이러한 관계를 보여준다. 따라서 의사결정 직후 상태를 사용하여 시스템 상태를 표현하면 크기가 $1/|\hat{M}_t|$배로 줄어든다. 앞에서 예로 든 문제에 대하여 의사 결정 직후 상태 S_t^x의 크기는 시스템 상태 S_t보다1/35배가 된다. 그러나, 상태의 크기가 축소되었지만 정확한 해를 구하기 어려울 정도로 여전히 크다.

둘째, MDP 모형에서는 가치 $J(S_t)$를 정확하게 계산하기 위해서는 식 (7)에 의거하여 S_t로부터 t시점이후 모든 시간에 대해 전이가 가능한 모든 상태의 가치를 정확하게 알고 있어야 한다. 그러므로 대부분의 DWTA 문제에서 $J(S_t)$를 정확하게 계산하는 것은 많은 시간이 소요된다. 이와는 대조적으로 강화학습 기반의 알고리즘에서는 가치를 근사적으로 구하는 함수를 사용하여 계산하며, 샘플로 모집단의 통계치를 추정할 때 샘플의 수를 조정하는 것처럼, 가치 함수를 추정할 때 입력으로 사용하는 S_t의 샘플의 크기를 사용자가 조절할 수 있다. 이와 같이 $|S_t|$보다 훨씬 적은 샘플을 취하여 계산함으로써 매우 빠른 시간 내에 근사치 추정이 가능해진다.

의사결정 직후 상태의 가치를 $J^x(S_t^x)$라 하면 상태의 가치와 의사결정 직후 상태의 가치의 관계는 다음과 같다 [20, 21].

$$J(S_t) = \min_{x_t \in X_t} (C(S_t, x_t) + \gamma J^x(S_t^x)) \quad (11)$$

$$J^x(S_t^x) = E[J(S_{t+1}) \mid S_t^x] \quad (12)$$

상태의 가치와 의사결정 직후 상태의 가치의 관계를 나타내는 식 (11)을 $J^x(S_{t-1}^x)$에 대입하여 계산하면 의사결정 직후 상태에 대한 다음과 같은 방정식을 얻는다.

$$J^x(S_{t-1}^x) = E\left[\min_{x_t \in X_t} (C(S_t, x_t) + \gamma J^x(S_t^x)) | S_{(t-1)}^x \right] \quad (13)$$

이 식에서 J가 선형 기저 함수에 의해 결정된다고 가정하자. 이 가정 하에서 의사결정 직후 상태의 가치를 계산하면, 정확한 값을 추정하지는 못하지만, 빠른 시간 안에 근사해를 구하는 장점을 가진다. 의사결정 직후 상태의 가치를 추정하기 위하여 선택된 기저 함수를 $\phi_f(S_t)$, $f \in F$라 하자. 이때, 기저 함수들의 집합 F는 시스템 변수의 차원을 $|F|$로 줄여 관심있는 요소들로 표현할 수 있다. 그러면, 의사결정 직후 상태의 가치를 $(\bar{J}^x)(S_t^x)$라 할 때, 다음과 같이 선형식으로 나타낼 수 있다.

$$(\bar{J}^x)(S_t^x) = \sum_{f \in F} \theta_f \phi_f(S_t^x) \quad (14)$$

의사결정 직후 상태의 가치를 나타내는 위 식에서 θ_f는 파라미터 벡터다. 식 (13)에서 $J^x(S_t^x)$ 및 $J^x(S_{t-1}^x)$ 대신에 의사결정 직후 상태의 가치를 나타내는 선형 회귀식 (14)를 사용하여 다시 쓰면 다음과 같다.

$$\bar{J}^x(S_{t-1}^x) = E\left[\min_{x_t \in X_t} (C(S_t, x_t) + \gamma \sum_{f \in F} \theta_f \phi_f(S_t^x)) | S_{t-1}^x \right] \quad (15)$$

강화학습에서는 API(Approximate Policy Iteration)라는 전략을 통하여 식 (7)의 Bellman 방정식에 대한 해로서 식 (8)을 사용하는 대신에, 의사결정 직후 상태의 가치를 나타내는 식 (15)를 이용한 시뮬레이션을 통하여 근사해를 얻는다. API 전략은 정책 평가를 정해진 수만큼 반복한 후, 그 결과를 정책 개선에 반영하는 방법을 반복적으로 실행한다. 강화학습의 API 중에 LSTD(Least Square Time Difference) 알고리즘과 LSPE(Least Square Policy Evaluation) 알고리즘은 서로 비슷한 성능을 낸다고 알려져 있으므로[2, 3], 본 논문에서는 이 두 알고리즘을 사용하여 해를 구하고, 그 성능을 비교한다.

<Figure 2>의 API-LSTD 알고리즘은 정책 개선을 N번 실행하며, 각 정책 개선 루프 안에 K번의 정책 평가를 한다. 먼저 θ^0를 초기화한 후에 정책 평가 루프에서 샘플링된 K개의 의사 결정 직후 상태에 대하여 기저 함수 $\phi(S_{t-1,1}^x)$ 및 $\phi(S_{t,1}^x)$와 Cost를 기록한다. 정책 개선 루프에서는, 기록된 기저 함수 행렬과 Cost 벡터를 이용하여 θ^1를 계산한다. θ^1을 이용하여 정책 평가 루프를 실행한 후 θ^2를 구한다. 이런 방법으로 정책 개선 루프를 N번 실행하여 θ^N 을 구한 다음 $X^\pi(S_t|\theta^N)$을 계산함으로써 알고리즘을 마치게 된다. 각 알고리즘의 정책 개선 루프의 마지막 행에서 새로운 θ를 구하기 위한 수식에서 사용된 기저 함수 행렬과 Cost 벡터는 다음과 같다.

1: Step 0 : Initialize θ
2: Step 1 :
3: for n = 1 to N (Policy Improvement Loop)
4: Step 2:
5: for k = 1 to K (Policy Evaluation Loop)
6: Generate a random post-decision state $S^x_{t-1,k}$
7: Record $\phi(S^x_{t-1,k})$
8: Simulate $S_{t,k}$ using Equation (6).
9: Determine $x_{t,k} = X^\pi(S_{t,k}|\theta^{n-1})$ using Equation (5), (7), (9)
10: Record $C(S_{t,k}, x_{t,k})$
11: Record $S^x_{t,k}$ using $x_{t,k}$ and Equation (5)
12: Record $\phi(S^x_{t,k})$
13: end for
14: Update θ^n and the policy:
15: $\hat{\theta} = [(\Phi_{t-1} - \gamma\Phi_t)^T(\Phi_{t-1} - \gamma\Phi_t)]^{-1}(\Phi_{t-1} - \gamma\Phi_t)^T C_t$
16: $\theta^n = \alpha_n\hat{\theta} + (1-\alpha_n)\theta^{n-1}$
17: end for
18: Return $X^\pi(S_t|\theta^N)$ and θ^N
19: End

<Figure 2> API-LSTD Algorithm [7, 17, 24]

$$\Phi_{t-1} \triangleq \begin{bmatrix} \phi(S^x_{t-1,1})^\top \\ \vdots \\ \phi(S^x_{t-1,K})^\top \end{bmatrix}, \Phi_t \triangleq \begin{bmatrix} \phi(S^x_{t,1})^\top \\ \vdots \\ \phi(S^x_{t,K})^\top \end{bmatrix}, C_t \triangleq \begin{bmatrix} C(S_{t,1}, x_t) \\ \vdots \\ C(S_{t,K}, x_t) \end{bmatrix} \quad (16)$$

<Figure 3>의 API-LSPE 알고리즘은 정책 개선 루프와 각 정책 개선 루프 안에 정책 평가를 정해진 횟수만큼 반복한다는 점에서는 API-LSTD 알고리즘과 동일하다. 차이점은 LSTD 알고리즘은 근사함수에 사용하는 기저함수에서 한 시점의 의사결정 직후 상태와 그 다음 시점의 의사결정 직후 상태를 이용하는 반면에, LSPE 알고리즘은 기저 함수가 한 시점의 의사결정 직후 상태를 사용한다는데 있다.

1: Step 0 : Initialize
2: Step 1 :
3: for n = 1 to N (Policy Improvement Loop)
4: Step 2:
5: for k = 1 to K (Policy Evaluation Loop)
6: Generate a random $S^x_{t-1,k}$
7: Record $\phi(S^x_{t-1,k})$
8: Simulate $S_{t,k}$ using Equation (6).
9: Determine $x_{t,k} = X^\pi(S_{t,k}|\theta^{n-1})$ using Equation (5), (7), (9)
10: Record $C(S_{t,k}, x_{t,k})$
11: end for
12: Update θ^n and the policy:
13: $\hat{\theta} = [(\Phi_{t-1})^T(\Phi_{t-1})]^{-1}(\Phi_{t-1})^T C_t$
14: $\theta^n = \alpha_n\hat{\theta} + (1-\alpha_n)\theta^{n-1}$
15: end for
16: Return $X^\pi(S_t|\theta^N)$ and θ^N
17: End

<Figure 3> API-LSPE Algorithm [24]

4. 시뮬레이션 및 결과분석

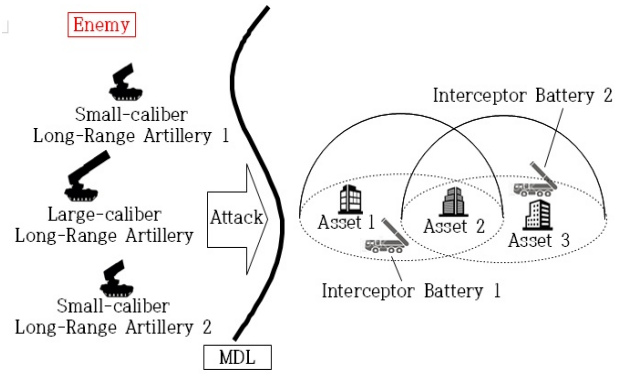
4.1 시뮬레이션 시나리오

북한은 아군의 대응에 혼란을 주기 위하여 정점고도가 다른 여러 종류의 방사포를 섞어 쏘는 전술을 구사한다. 따라서 이러한 경우에 적 장사정포 공격시 API-LSTD 및 API-LSPE 알고리즘을 활용한 표적할당의 계산시간이 단시간에 이루어지는지 시뮬레이션을 통하여 확인하며, 계산 결과로 얻어진 근사해가 우수한 해인지를 판단하기 위하여 기존의 SSL 발사방법을 적용한 해와 비교한다.

<Figure 4>에 적군 및 아군의 배치가 요약되어 있다. 아군은 3개의 자산을 방어하며, 각 자산의 가치는 (1, 10, 5)로 가정한다. 요격 포대는 2개의 장소에 배치되어 있으며, 포대 1은 자산 1, 2를 방어하고 포대 2는 자산 2, 3을 방어한다. 요격 포대는 한 번의 방어에 최대 4회의 유도탄을 사용할 수 있으며, 각 요격 포대는 적 장사정포 포대의 1회 공격에 대하여 최대 2회 방어한다고 가정한다. 포탄 1 단위는 12발로 한다. 1회의 요격 유도탄은 소구경 장사정포 1회(24발)공격에 대하여 포탄 2 단위를 사용하여 요격하며, 대구경 장사정포 1회(12발)에 공격에 대하여 포탄 1 단위를 사용하여 요격한다. 대구경 장사정포에 대한 유도탄의 명중률은 70%이며, 소구경 장사정포에 대한 유도탄의 명중률은 90%이다.

장사정포 공격은 한 번의 공격에 최대 3문의 장사정포를 사용하여 공격할 수 있다. 소구경 장사정포는 2문을 보유하고 있으며, 한 번 공격에 각 24발을 발사하며, 1 발이 자산에 떨어질 때 마다 최초 가치의 2.5%씩 자산의 가치가 감소한다. 대구경 장사정포는 1문을 보유하고 있으며, 한 번 공격에 12발을 발사하며, 1 발이 자산에 떨어지면 최초 가치의 5% 만큼 자산의 가치가 감소한다. 각 장사정포의 1회 공격시 하나의 자산을 향하여 공격한다. 한 번의 공격에 동원하는 장사정포의 문 수는 1, 2, 3문중 하나이며 동일한 확률로 사용한다. 적군은 아군 자산의 잔존 가치에 따른 다항분포에 의거하여 자산을 선택하여 공격한다. 예를 들어 자산의 잔존가치가 (1, 8, 5)이면 (1/14, 8/14, 5/14)의 매개변수를 갖는 다항분포에 의하여 공격할 자산을 선택하며, 자산의 잔존가치가 (0.5, 5.5, 2)이면 (0.5/8, 5.5/8, 2/8)의 매개변수를 갖는 다항분포에 의하여 자산을 선택하여 공격한다.

LSTD 및 LSPE 알고리즘에서 시점에서의 의사결정 직후 상태의 가치를 추정할 때 사용하는 기저함수로는 $\phi_0(S^x_t) = 1$, $\phi_1(S^x_t) = A_t$, $\phi_2(S^x_t) = R^x_t$, $\phi_3(S^x_t) = A^x_t$를 사용하였으며, α는 1, γ는 0.8을 사용하였다. 두 알고리즘에서 정책평가는 매 정책개선 루프마다 50번 실행하며, 정책개선 루프는 15번 실행하였다.



<Figure 4> Scenario Layout

4.2 시뮬레이션 결과

본 연구에서는 요격 유도탄 재고에 따라 2개의 시나리오를 만들어 시뮬레이션을 실행하였다. 시나리오 1에서 = R_t(4,4), 시나리오 2에서 R_t = (8,8)로 설정하였다. 모든 경우 A_t = (1,1,1)로 하였다. 모든 시나리오에서 두 알고리즘으로 시뮬레이션을 실행하면 1초 이내에 결과를 얻었다.

4.2.1 시나리오 1: R_t = (4,4)

<Table 1>은 아군의 2개 방어포대에 각 4단위(48발)의 유도탄 재고가 있는 경우, 공격 벡터 별로 두 알고리즘을 적용한 의사결정 결과와 SSL 발사방법을 적용한 결과를 보여준다. <Table 1>에서 공격 벡터는 6자리 숫자로 구성되어 있다. 6자리 숫자 중 첫 3자리 숫자는 대구경 장사정포가 자산 1, 2, 3에 대하여 공격하는지 여부를 1과 0으로 나타내며, 나머지 3자리 숫자는 소구경 장사정포가 자산 1, 2, 3에 대하여 몇 회 공격하는지를 나타낸다. 이 시나리오를 적용하여 시뮬레이션을 한 결과, API-LSTD 알고리즘을 적용 했을 때의 자산가치 손실이 SSL 발사방법을 적용 했을 때의 자산 가치의 손실보다, 공격 벡터의 공격 확률을 고려한 가중평균치가 18.6% 작았다. API-LSPE 알고리즘을 적용한 경우, SSL 발사방법을 사용하는 경우보다 자산 손실의 가중평균이 17.9% 작았다. API-LSTD 알고리즘을 적용한 경우, API-LSPE 알고리즘을 적용한 경우보다 자산 손실이 0.9% 작았다. 따라서, 시나리오 1에 대한 두 알고리즘의 우열은 가릴 수 없다. 아울러 어떤 종류의 공격 벡터에 대해서도 강화학습 기반의 알고리즘들은 SSL 발사방법에 비하여 자산 손실이 작은 결과를 가져왔다.

<Table 1> Policy Comparison for the Scenario 1

Attack Probability (%)	Attack Vector	LSTD Policy		LSTD Value ①	SSL Value ②	LSPE Value ③	LSPE Policy		GAP (%)		
		Battery 1	Battery 2				Battery 1	Battery 2	(②−①)/②	(②−③)/②	(③−①)/③
7.41	0 0 0 0 0 1	0 0 0 0 0 0	0 0 0 1 0 0	8.30	9.94	**8.30**	0 0 0 0 0 0	0 0 0 1 0 0	16.5	16.5	0.0
1.24	0 0 0 0 0 2	0 0 0 0 0 0	0 0 0 1 1 0	9.26	12.45	**9.18**	0 0 0 0 0 0	0 0 0 1 1 0	25.6	26.2	-0.8
7.41	0 0 0 0 1 0	0 0 0 1 0 0	0 0 0 0 0 0	**8.38**	9.81	8.39	0 0 0 1 0 0	0 0 0 0 0 0	14.5	14.5	0.1
2.47	0 0 0 0 1 1	0 0 0 1 0 0	0 0 0 0 1 0	9.36	11.03	**9.32**	0 0 0 1 0 0	0 0 0 0 1 0	15.1	15.5	- 0.5
1.24	0 0 0 0 2 0	0 0 0 1 1 0	0 0 0 0 0 0	9.17	12.99	**9.14**	0 0 0 1 1 0	0 0 0 0 0 0	29.4	29.7	- 0.4
7.41	0 0 0 1 0 0	0 0 0 0 0 0	0 0 0 0 0 0	**7.93**	9.79	8.14	0 0 0 1 0 0	0 0 0 0 0 0	19.0	16.9	2.6
2.47	0 0 0 1 0 1	0 0 0 0 0 0	0 0 0 0 1 0	**8.78**	11.01	9.10	0 0 0 1 0 0	0 0 0 0 1 0	20.3	17.4	3.5
2.47	0 0 0 1 1 0	0 0 0 0 1 0	0 0 0 0 0 0	**8.86**	11.02	9.18	0 0 0 1 1 0	0 0 0 0 0 0	19.6	16.7	3.5
1.24	0 0 0 2 0 0	0 0 0 0 0 0	0 0 0 0 0 0	7.93	10.54	**7.91**	0 0 0 0 0 0	0 0 0 0 0 0	24.8	24.9	- 0.2
3.70	0 0 1 0 0 0	0 0 0 0 0 0	0 0 1 0 0 0	8.36	10.81	**8.29**	0 0 0 0 0 0	0 0 2 0 0 0	22.7	23.2	- 0.7
2.47	0 0 1 0 0 1	0 0 0 0 0 0	0 0 1 0 1 0	**9.26**	10.18	9.33	0 0 0 0 0 0	0 0 2 0 1 0	9.1	8.4	0.7
1.24	0 0 1 0 0 2	0 0 0 0 0 0	0 0 0 0 0 0	10.82	11.05	**10.78**	0 0 0 0 0 0	0 0 2 0 1 0	5.9	6.2	- 0.3
2.47	0 0 1 0 1 0	0 0 0 0 1 0	0 0 1 0 0 0	**9.28**	11.84	9.31	0 0 0 0 1 0	0 0 2 0 0 0	21.6	21.4	0.3
2.47	0 0 1 0 1 1	0 0 0 0 1 0	0 0 1 0 0 1	**10.21**	11.26	10.28	0 0 0 0 1 0	0 0 2 0 0 1	9.3	8.7	0.7
1.24	0 0 1 0 2 0	0 0 0 0 1 1	0 0 1 0 0 0	**10.27**	12.51	10.33	0 0 0 0 1 1	0 0 2 0 0 0	17.9	17.4	0.5
2.47	0 0 1 1 0 0	0 0 0 0 0 0	0 0 1 0 0 0	**8.84**	11.83	9.09	0 0 0 0 1 0	0 0 2 0 0 0	25.3	23.1	2.8
2.47	0 0 1 1 0 1	0 0 0 0 0 0	0 0 1 0 0 1	**9.72**	11.24	10.09	0 0 0 0 1 0	0 0 2 0 0 1	13.6	10.2	3.7
2.47	0 0 1 1 1 0	0 0 0 0 0 1	0 0 1 0 0 0	**9.74**	12.49	10.12	0 0 0 0 1 1	0 0 2 0 0 0	22.0	19.0	3.8
1.25	0 0 1 2 0 0	0 0 0 0 0 0	0 0 1 0 0 0	8.84	12.23	**8.74**	0 0 0 0 0 0	0 0 2 0 0 0	27.7	28.5	-1.1
3.70	0 1 0 0 0 0	0 2 0 0 0 0	0 0 0 0 0 0	8.39	12.07	**8.38**	0 2 0 0 0 0	0 0 0 0 0 0	30.5	30.5	0.0
2.47	0 1 0 0 0 1	0 2 0 0 0 0	0 0 0 0 1 0	9.36	12.72	**9.31**	0 2 0 0 0 0	0 0 0 0 1 0	26.4	26.8	-0.5
1.24	0 1 0 0 0 2	0 2 0 0 0 0	0 0 0 0 1 1	**10.28**	13.85	10.28	0 2 0 0 0 0	0 0 0 0 1 1	25.8	25.8	0.1
2.47	0 1 0 0 1 0	0 2 0 0 1 0	0 0 0 0 0 0	9.44	10.60	**9.40**	0 2 0 0 1 0	0 0 0 0 0 0	11.0	11.3	-0.4

<Table 1> Policy Comparison for the Scenario 1(Continued)

Attack Probability (%)	Attack Vector	LSTD Policy		LSTD Value ①	SSL Value ②	LSPE Value ③	LSPE Policy		GAP (%)		
		Battery 1	Battery 2				Battery 1	Battery 2	②−① / ②	②−③ / ②	③−① / ③
2.47	0 1 0 0 1 1	0 2 0 0 1 0	0 0 0 0 0 1	10.35	11.93	**10.32**	0 2 0 0 1 0	0 0 0 0 0 1	13.2	13.5	-0.2
1.24	0 1 0 0 2 0	0 2 0 0 1 0	0 0 0 0 0 1	10.47	11.37	**10.43**	0 2 0 0 1 0	0 0 0 0 0 0	7.9	8.3	-0.4
2.47	0 1 0 1 0 0	0 2 0 0 0 0	0 0 0 0 0 0	**8.86**	12.31	9.18	0 2 0 0 1 0	0 0 0 0 0 0	28.0	25.4	3.4
2.47	0 1 0 1 0 1	0 2 0 0 0 0	0 0 0 0 0 1	**9.82**	13.02	10.12	0 2 0 0 1 0	0 0 0 0 0 1	24.6	22.3	3.0
2.47	0 1 0 1 1 0	0 2 0 0 0 1	0 0 0 0 0 0	9.90	11.23	**9.85**	0 2 0 0 0 1	0 0 0 0 0 0	11.8	12.2	-0.4
1.25	0 1 0 2 0 0	0 2 0 0 0 0	0 0 0 0 0 0	8.86	12.67	**8.84**	0 2 0 0 0 0	0 0 0 0 0 0	30.0	30.2	-0.2
3.70	1 0 0 0 0 0	0 0 0 0 0 0	0 0 0 0 0 0	7.97	9.85	**7.94**	1 0 0 0 0 0	0 0 0 0 0 0	19.1	19.4	-0.5
2.47	1 0 0 0 0 1	0 0 0 0 0 0	0 0 0 0 1 0	**8.82**	10.79	8.82	1 0 0 0 0 0	0 0 0 0 1 0	18.3	18.2	0.1
1.24	1 0 0 0 0 2	0 0 0 0 0 0	0 0 0 0 1 1	9.91	12.20	**9.77**	1 0 0 0 0 0	0 0 0 0 1 1	18.8	19.9	-1.5
2.47	1 0 0 0 1 0	0 0 0 0 1 0	0 0 0 0 0 0	**8.90**	10.79	8.95	1 0 0 0 1 0	0 0 0 0 0 0	17.5	17.1	0.6
2.47	1 0 0 0 1 1	0 0 0 0 1 0	0 0 0 0 0 1	9.87	12.10	**9.83**	1 0 0 0 1 0	0 0 0 0 0 1	18.5	18.8	-0.3
1.24	1 0 0 0 2 0	0 0 0 0 1 1	0 0 0 0 0 0	**9.94**	11.59	9.99	1 0 0 0 0 1	0 0 0 0 1 0	14.3	13.9	0.5
2.47	1 0 0 1 0 0	0 0 0 0 0 0	0 0 0 0 0 0	**8.35**	9.87	8.70	1 0 0 0 1 0	0 0 0 0 0 0	15.4	11.8	4.0
2.47	1 0 0 1 0 1	0 0 0 0 0 0	0 0 0 0 0 1	**9.18**	11.09	9.61	1 0 0 0 1 0	0 0 0 0 0 0	17.2	13.3	4.5
2.47	1 0 0 1 1 0	0 0 0 0 0 1	0 0 0 0 0 0	**9.26**	11.09	9.41	1 0 0 0 01	0 0 0 0 0 0	16.5	15.1	1.6
1.24	1 0 0 2 0 0	0 0 0 0 0 0	0 0 0 0 0 0	8.35	10.28	**8.33**	1 0 0 0 1 1	0 0 0 0 0 0	18.8	19.0	-0.3

4.2.2 시나리오 2: R_t = (8,8)

<Table 2>는 아군의 2개 방어 포대에 각 8 단위(48발)의 유도탄 재고가 있는 경우, 장사정포의 공격 벡터에 따른 두 알고리즘을 적용한 의사결정 결과와 SSL 발사방법을 적용한 결과를 보여준다. API-LSTD 알고리즘을 적용 했을 때의 자산가치 손실이 SSL 발사방법을 적용 했을 때의 자산 가치의 손실보다 평균 27.7% 작았다. API-LSPE 알고리즘을 적용한 경우, SSL 발사방법을 취하는 경우보다 손실이 평균 22.4% 작았다. API-LSTD 알고리즘을 적용한 경우, API-LSPE 알고리즘을 적용한 경우보다 자산 손실이 평균 6% 작았다. 따라서, 시나리오 2에 대해서는 API-LSTD 알고리즘이 API-LSPE 알고리즘을 적용하는 경우보다 우수한 결과를 가져왔다. 아울러 어떤 종류의 공격 벡터에 대해서도 강화학습 기반의 알고리즘은 SSL 발사방법보다 자산 손실이 작은 결과를 가져왔다.

<Table 2> Policy Comparison for the Scenario 2

Attack Probability (%)	Attack Vector	LSTD Policy		LSTD Value ①	SSL Value ②	LSPE Value ③	LSPE Policy		GAP (%)		
		Battery 1	Battery 2				Battery 1	Battery 2	②−① / ②	②−③ / ②	③−① / ③
7.41	0 0 0 0 0 1	0 0 0 0 0 0	0 0 0 1 0 0	**6.19**	8.41	6.82	0 0 0 0 0 0	0 0 0 1 0 0	26.4	18.9	9.2
1.24	0 0 0 0 0 2	0 0 0 0 0 0	0 0 0 1 1 0	**6.62**	10.25	7.25	0 0 0 0 0 0	0 0 0 1 1 0	35.4	29.3	8.7
7.41	0 0 0 0 1 0	0 0 0 1 0 0	0 0 0 0 0 0	**6.35**	8.31	7.01	0 0 0 2 0 0	0 0 0 0 0 0	23.6	15.6	9.4
2.47	0 0 0 0 1 1	0 0 0 1 0 0	0 0 0 0 1 0	**6.95**	9.06	7.70	0 0 0 2 0 0	0 0 0 0 1 0	23.3	15.0	9.7
1.24	0 0 0 0 2 0	0 0 0 1 1 0	0 0 0 0 0 0	**6.73**	11.91	10.95	0 0 0 2 2 0	0 0 0 0 0 0	43.5	8.1	38.5
7.41	0 0 0 1 0 0	0 0 0 0 0 0	0 0 0 0 0 0	**6.14**	8.29	6.59	0 0 0 1 0 0	0 0 0 0 0 0	25.9	20.5	6.8
2.47	0 0 0 1 0 1	0 0 0 0 0 0	0 0 0 0 1 0	**6.70**	9.03	7.22	0 0 0 1 0 0	0 0 0 0 1 0	25.8	20.0	7.2
2.47	0 0 0 1 1 0	0 0 0 0 1 0	0 0 0 0 0 0	**6.86**	9.05	7.48	0 0 0 1 2 0	0 0 0 0 0 0	24.2	17.3	8.3
1.24	0 0 0 2 0 0	0 0 0 0 0 0	0 0 0 0 0 0	**6.13**	9.03	6.98	0 0 0 1 1 0	0 0 0 0 0 0	32.1	22.7	12.2
3.70	0 0 1 0 0 0	0 0 0 0 0 0	0 0 1 0 0 0	**6.50**	9.78	6.81	0 0 0 0 0 0	0 0 2 0 0 0	33.5	30.4	4.6
2.47	0 0 1 0 0 1	0 0 0 0 0 0	0 0 1 0 1 0	**7.06**	8.79	7.44	0 0 0 0 0 0	0 0 2 0 1 0	19.7	15.4	5.1
1.24	0 0 1 0 0 2	0 0 0 0 0 0	0 0 1 0 1 1	**7.70**	10.27	8.16	0 0 0 0 0 0	0 0 2 0 1 1	25.0	20.5	5.6
2.47	0 0 1 0 1 0	0 0 0 0 1 0	0 0 1 0 0 0	**7.22**	10.31	7.70	0 0 0 0 2 0	0 0 2 0 0 0	30.0	25.3	6.2
2.47	0 0 1 0 1 1	0 0 0 0 1 0	0 0 1 0 0 1	**7.81**	9.46	8.37	0 0 0 0 2 0	0 0 2 0 0 1	17.4	11.5	6.7
1.24	0 0 1 0 2 0	0 0 0 0 1 1	0 0 1 0 0 0	**7.97**	10.86	8.64	0 0 0 0 2 2	0 0 2 0 0 0	26.6	20.4	7.8
2.47	0 0 1 1 0 0	0 0 0 0 0 0	0 0 1 0 0 0	**7.01**	10.29	7.22	0 0 0 0 1 0	0 0 2 0 0 0	31.9	28.9	2.9
2.47	0 0 1 1 0 1	0 0 0 0 0 0	0 0 1 0 0 1	**7.57**	9.44	7.91	0 0 0 0 1 0	0 0 2 0 0 1	19.8	16.2	4.3

<Table 2> Policy Comparison for the Scenario 2 (Continued)

Attack Probability (%)	Attack Vector	LSTD Policy		LSTD Value ①	SSL Value ②	LSPE Value ③	LSPE Policy		GAP (%)		
		Battery 1	Battery 2				Battery 1	Battery 2	(②−①)/②	(②−③)/②	(③−①)/③
2.47	0 0 1 1 1 0	0 0 0 0 0 1	0 0 1 0 0 0	**7.73**	10.84	8.18	0 0 0 0 1 2	0 0 2 0 0 0	28.7	24.5	5.5
1.25	0 0 1 2 0 0	0 0 0 0 0 0	0 0 1 0 0 0	**7.01**	10.59	7.53	0 0 0 0 1 1	0 0 2 0 0 0	33.8	28.9	6.9
3.70	0 1 0 0 0 0	0 2 0 0 0 0	0 0 0 0 0 0	**6.34**	11.30	6.91	0 2 0 0 0 0	0 0 0 0 0 0	43.9	38.8	8.2
2.47	0 1 0 0 0 1	0 2 0 0 0 0	0 0 0 0 1 0	**6.94**	11.74	7.52	0 2 0 0 0 0	0 0 0 0 1 0	40.9	35.9	7.7
1.24	0 1 0 0 0 2	0 2 0 0 0 0	0 0 0 0 1 1	**7.61**	12.09	8.19	0 2 0 0 0 0	0 0 0 0 1 1	37.1	32.3	7.1
2.47	0 1 0 0 1 0	0 2 0 0 1 0	0 0 0 0 0 0	**7.10**	8.87	7.77	0 2 0 0 2 0	0 0 0 0 0 0	20.0	12.4	8.6
2.47	0 1 0 0 1 1	0 2 0 0 1 0	0 0 0 0 0 1	**7.34**	9.69	8.45	0 2 0 0 2 0	0 0 0 0 0 1	24.3	12.8	13.1
1.24	0 1 0 0 2 0	0 2 0 0 1 1	0 0 0 0 0 0	**7.89**	9.71	8.74	0 0 0 0 2 2	0 2 0 0 0 0	18.7	10.0	9.7
2.47	0 1 0 1 0 0	0 2 0 0 0 0	0 0 0 0 0 0	**6.86**	11.60	7.30	0 2 0 0 1 0	0 0 0 0 0 0	40.9	37.1	6.0
2.47	0 1 0 1 0 1	0 2 0 0 0 0	0 0 0 0 0 1	**7.45**	12.09	7.98	0 2 0 0 1 0	0 0 0 0 0 1	38.4	34.0	6.6
2.47	0 1 0 1 1 0	0 2 0 0 0 1	0 0 0 0 0 0	**7.61**	9.68	8.25	0 1 0 0 1 2	0 1 0 0 0 0	21.4	14.8	7.8
1.25	0 1 0 2 0 0	0 2 0 0 0 0	0 0 0 0 0 0	**6.86**	12.04	7.59	0 2 0 0 1 1	0 0 0 0 0 0	43.0	37.0	9.6
3.70	1 0 0 0 0 0	0 0 0 0 0 0	0 0 0 0 0 0	**6.18**	8.51	6.20	1 0 0 0 0 0	0 0 0 0 0 0	27.4	27.1	0.3
2.47	1 0 0 0 0 1	0 0 0 0 0 0	0 0 0 0 1 0	**6.75**	9.17	6.85	1 0 0 0 0 0	0 0 0 0 1 0	26.4	25.3	1.5
1.24	1 0 0 0 0 2	0 0 0 0 0 0	0 0 0 0 1 1	**7.38**	9.57	7.57	1 0 0 0 0 0	0 0 0 0 1 1	22.9	20.9	2.5
2.47	1 0 0 0 1 0	0 0 0 0 1 0	0 0 0 0 0 0	**6.91**	9.06	6.94	1 0 0 0 2 0	0 0 0 0 0 0	23.7	23.4	0.4
2.47	1 0 0 0 1 1	0 0 0 0 1 0	0 0 0 0 0 1	**7.50**	9.89	7.64	1 0 0 0 2 0	0 0 0 0 0 1	24.2	22.8	1.8
1.24	1 0 0 0 2 0	0 0 0 0 1 1	0 0 0 0 0 0	**7.66**	9.90	8.07	1 0 0 0 2 1	0 0 0 0 0 1	22.6	18.5	5.1
2.47	1 0 0 1 0 0	0 0 0 0 0 0	0 0 0 0 0 0	**6.59**	8.60	**6.55**	1 0 0 0 1 0	0 0 0 0 0 0	23.4	23.8	-0.6
2.47	1 0 0 1 0 1	0 0 0 0 0 0	0 0 0 0 0 1	**7.14**	9.47	7.24	1 0 0 0 1 0	0 0 0 0 0 0	24.6	23.5	1.4
2.47	1 0 0 1 1 0	0 0 0 0 0 1	0 0 0 0 0 0	**7.31**	9.48	7.42	1 0 0 0 1 2	0 0 0 0 0 0	22.9	21.7	1.5
1.24	1 0 0 2 0 0	0 0 0 0 0 0	0 0 0 0 0 0	**6.59**	9.03	6.94	1 0 0 0 1 1	0 0 0 0 0 0	27.0	23.1	5.0

4.3 시뮬레이션 분석

이상의 두 시나리오에 대한 시뮬레이션의 결과는 다음과 같이 요약 할 수 있다.

첫째, MDP의 경우 주어진 문제의 크기가 클수록 최적화를 위해 기하급수적으로 많은 시간이 소요되는 것으로 알려져 있으나, 강화학습 기반의 알고리즘을 사용할 경우 매우 빠른 시간 안에 근사 해를 구할 수 있음을 확인하였다.

둘째, 강화학습 기반의 알고리즘을 적용하는 경우 공격 벡터 및 유도탄의 재고에 따라 요격 전략을 유연하게 사용하므로, 공격 벡터 및 유도탄의 재고 수준에 관계없이 고정적으로 대응하는 SSL 발사방법보다 자산 가치의 손실이 항상 작았다.

셋째, 두 시나리오에 대한 알고리즘을 적용한 결과, API-LSTD 알고리즘을 적용한 경우가 API-LSPE 알고리즘을 적용한 경우보다 자산 손실이 작은 결과를 가져왔다. 이는 의사결정 직후 상태의 가치를 추정할 때 사용하는 기저함수 행렬이 API-LSTD 알고리즘은 두 시점을 사용하고, API-LSPE 알고리즘을 한 시점의 기저함수 행렬을 사용하기 때문인 것으로 보인다.

5. 결 론

본 논문에서는 파괴력과 요격률이 다른 여러 구경의 장사정포 공격에 대하여 무기를 할당하는 문제에 강화학습 기반의 API-LSTD 및 API-LSPE 알고리즘을 적용하여 해를 구하는 시뮬레이션을 수행하였다. 두 알고리즘은 매우 짧은 시간에 복잡한 MDP 문제의 근사해를 산출하였으며, 산출된 근사해가 SSL 발사방법을 적용하여 구한 해보다 훨씬 우수하여 여러 구경의 장사정포에 의한 공격의 대응에 효과적으로 사용할 수 있음을 확인하였다.

이로써 본 논문의 학문적 공헌은 적용이 검증된 방법론을 장사정포 전체 위협 도메인에 적용을 통하여 적용범위를 확대하여도 좋은 방법론임을 검증하였고, 실제사례와 근접한 시뮬레이션을 보여준데 있다.

본 연구의 결과를 장사정포 위협에 대비한 방어 요격체계 개발에 적용한다면 기존의 SSL 발사방법보다 효과적인 방공망 설계에 기여할 수 있을 것으로 판단된다. 추후 연구 과제로, 요격 포대의 수, 각 포대별 커버리지 범위 및 유도탄 재고 수준, 방어 자산의 가치 등의 값을 바꾸어 가며 시뮬레이션을 한다면, 장사정포 공격의 대비에 필요한 여러 자원의 양적 및 질적 수준 확보에 도움이 될 것으

로 보인다. 아울러, 다양한 탄도탄을 섞어 쏘는 공격에 대비한 다층 방어망을 구축할 때 방어의 효과를 검증하기 위한 도구로서 본 논문의 방법론을 확장하여 사용 가능할 것으로 추정된다.

Acknowledgement

This study was partially supported by 2022 Industry-Academic Research Program of Hannam University and Hanwha systems.

References

[1] Bertsekas, D., Homer, M., Logan, D., Patek, S., and Sandell, N., Missile Defense and Interceptor Allocation by Neuro-Dynamic Programming, *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 2000, Vol. 30, No. 1, pp. 42-51.

[2] Bertsekas, D.P., Approximate Policy Iteration: A Survey and Some New Methods, *Journal of Control Theory and Applications*, 2011, Vol. 9, pp. 310-335.

[3] Bertsekas, D.P., *Dynamic Programming and Optimal Control: Approximate Dynamic Programming*, 4th Edition, 2012, Athena Scientific, Belmont, MA.

[4] Cha, Y.H. and Jeong, B., Exact Algorithm for the Weapon Target Assignment and Fire Scheduling Problem, *Journal of the Society of Korea Industrial and Systems Engineering*, 2019, Vol. 42, No. 1, pp. 143- 150.

[5] Choi, B.W., Yoo, B.C., Kim, J.H., and Yim, D.S., A Study on the Flight Trajectory Prediction Method of Ballistic Missiles, *2020 Autumn Conference on Journal of Korean Society of Systems Engineering*, 2020, pp. 131-140.

[6] Chosun Media, https://www.chosun.com/ politics/ diplomacy-defense/2022/04/10/3AQITBNGDZHBFGUI4WFTS3A4RA/ (accessed 2022/08/30)

[7] Davis, M.T., Robbins, M.J., and Lunday, B.J., Approximate Dynamic Programming for Missile Defense Interceptor Fire Control, *European Journal of Operational Research*, 2017, Vol. 259, pp. 873-886.

[8] Hong, D.W., Yim, D.S., and Choi, B.W., Application and Determination of Defended Footprint Using a Simulation Model for Ballistic Missile Trajectory, *Journal of the Korea Institute of Military Science and Technology*, 2018, Vol. 21, No. 4, pp. 551-561.

[9] Hosein, P.A., A Class of Dynamic Nonlinar Resource Allocation Problems, [Ph. D. Dissertation], Massachusetts Institute of Technology, 1989.

[10] Im, J.S., Yoo, B.C., Kim, J.H., and Choi, B.W., A Study of Multi-to-Majority Response on Threat Assessment and Weapon Assignment Algorithm: by Adjusting Ballistic Missiles and Long-Range Artillery Threat, *Journal of Korean Society of Industrial and systems Engineering*, 2021, Vol. 44, N0. 4, pp. 43-52.

[11] Jang, J.G., Kim, K., Choi, B.W., and Suh, J.J., A Linear Approximation Model for an Asset-based Weapon Target Assignment Problem, *Journal Society of Korea Industrial and System Engineering*, 2015, Vol. 38, No. 3, pp. 108-116.

[12] Jung, J.K., Uhm, H.S., and Lee, Y.H., Rolling-Horizon Scheduling Algorithm for Dynamic Weapon-Target Assignment in Air Defense Engagement, *Journal of the Korean Institute of Industrial Engineering*, 2020, Vol. 46, No. 1, pp. 11-24.

[13] Kim, J.H., Kim, K., Choi, B.W., and Suh, J.J., An Application of Quantum-inspired Genetic Algorithm for Weapon Target Assignment Problem, *Journal Society of Korea Industrial and System Engineering*, 2017, Vol. 40, No. 4, pp. 260-267.

[14] Kim, T., Yun, N., Kim, Y.J., Park, I., and Shim, D., Effect Analysis of Long-range Artillery Intercept System According to its Component Arrangement, *Journal of Korean Society Industrial and System Engineering*, 2022, Vol. 45, No. 1, pp. 41~52.

[15] Lee, C., Kim, J.-H., Choi, B.W., and Kim, K., Approximate Dynamic Programming Based Interceptor Fire Control and Effectiveness Analysis for M-to-M Engagement, *Journal of the Korean Society for Aeronautical & Space Science*, 2022, Vol. 50, No. 4, pp. 287-295.

[16] Lee, Z.J., Lee, C.Y., and Su, S.F., An Immunity Based Ant Colony Optimization Algorithm for Solving Weapon-Target Assignment Problem, *Applied Soft Computing*, 2002, Vol. 2, No. 1, pp. 39-47.

[17] McKenna, R.S., Robbins, M.J., Lunday, B.J., and McCormack, I.M., Approximate Dynamic Programming for the Military Inventory Routing Problem, *Annals of Operational Research*, 2020, Vol. 288, No. 1, pp. 391-416.

[18] Ministry of National Defense, 2020 Defense White Paper, 2020, Seoul, Ministry of National Defense.

[19] Naeem, H. and Masood, A., An Optimal Dynamic Threat Evaluation and Weapon Scheduling Technique, *Know-*

ledge-Based Systems, 2010, Vol. 23, No. 4, pp. 337-342.

[20] Powell, W.B., *Approximate Dynamic Programming: Solving the Curse of Dimensionality*, Second Edition, 2011, John Wiley & Sons, Hoboken, NJ.

[21] Powell, W. B., Perspectives of Approximate Dynamic Programming, *Annals of Operations Research*, 2012, Vol 13, No. 2, pp. 1-38.

[22] Rempel, M. and Bai, J., A Review of Approximate Dynamic Programming Applications within Military Operations Research, *Operational Research Perspectives*, 2021, Vol. 8, 100204.

[23] Shin, M.K., Park, S.-S., Lee, D., and Choi, H.-L., "Mean Field Game based Reinforcement Learning for Weapon-Target Assignment", *Journal of the Korea Institute of Military Science and Technology*, 2020, Vol. 23, No. 4, pp. 337-345.

[24] Summers, D.S., Robbins, M.J., and Lunday, B.J., An Approximate Dynamic Programming for Comparing Firing Policies in a Networked Air Defense Environment, *Computers & Operations Research*, 2020, Vol. 117, 104890.

[25] Yonhapnews, https://www.yna.co.kr/view/AKR20220410019151504 (accessed 2022/08/30)

[26] Yoo, B.C., Kim, J.H., Kwon, Y.S., and Choi, B.W., A Study on the Flight Trajectory Prediction Method of Ballistic Missiles, *Journal of Korean Society of Systems Engineering*, 2020, Vol. 16, No. 2, pp. 131-140.

ORCID

Hyeonho Kim | http://orcid.org/0000-0002-2518-3267
Jung Hun Kim | http://orcid.org/0000-0001-8139-3237
Joohoe Kong | http://orcid.org/0000-0001-7322-6180
Ji Hoon Kyung | http://orcid.org/0000-0002-0359-5594