Journal of the KIMST, Vol. 23, No. 4, pp. 381-388, 2020
ISSN 1598-9127(Print) · 2636-0640(Online)
DOI https://doi.org/10.9766/KIMST.2020.23.4.381

Research Paper 정보·통신 부문

무기할당을 위한 계층적 레이지 그리디 알고리즘

정 혜 선[*,1)]

1) 국방과학연구소 제1기술연구본부

Hierarchical Lazy Greedy Algorithm for Weapon Target Assignment

Hyesun Jeong[*,1)]

1) *The 1st Research and Development Institute, Agency for Defense Development, Korea*

(Received 20 March 2020 / Revised 8 July 2020 / Accepted 24 July 2020)

Abstract

Weapon target assignment problem is an essential technology for automating the operator's rapid decision-making support in a battlefield situation. Weapon target assignment problem is a kind of the optimization problem that can build up an objective function by maximizing the number of threat target destructed or maximizing the survival rate of the protected assets. Weapon target assignment problem is known as the NP-Complete, and various studies have been conducted on it. Among them, a greedy heuristic algorithm which guarantees $(1-1/e)$ approximation has been considered a very practical method in order to enhance the applicability of the real weapon system. In this paper, we formulated the weapon target assignment problem for supporting decision-making at the level of artillery. The lazy strategy based on hierarchical structure is proposed to accelerate the greedy algorithm. By experimental results, we show that our algorithm is more efficient in processing time and support the same level of the objective function value with the basic greedy algorithm.

Key Words : Lazy Greedy(레이지 그리디), Weapon Target Assignment(무기할당), Submodular Maximization(서브모듈라 최대화)

1. 서 론

무기할당은 다수 표적 교전 상황에서 운용자의 업무 로드를 경감시키기 위해서 표적에 대응하는 무장을 자동으로 추천하는 기술로, 운용자는 추천된 무기할당 결과를 확인하여 최종적인 의사결정을 내린다. 이러한 무기할당 기술은 신속 대응이 요구되는 대공방어에 필수적인 기술이기 때문에 표적의 종류 및 요격 사거리에 따른 대응 능력에 따라 다양한 유도무기 체계에 적용되고 있다.

무기할당 문제는 위협 표적에 방어 무기를 효율적

* Corresponding author, E-mail: ssun9674@naver.com

으로 할당하여 자산 방어율을 최대화하거나 위협 표적 타격률을 최대화하는 목적을 실현하는 것으로 최적화 문제 부류에 속한다. 무기할당 문제는 타격 표적 최대화하는 표적 기반 무기할당과 방어 자산의 생존을 최대화하는 자산 기반 무기할당 방법이 있다. 무기할당은 가능한 모든 표적-무기 쌍으로 이루어진 집합의 가능한 모든 부분집합 중에서 목적함수가 최대인 부분집합을 찾는 문제인 조합 최적화(Combinatorial Optimization)문제에 속한다. 조합 최적화 문제는 이산변수가 존재하기 때문에 심플렉스(Simplex)와 같이 잘 알려진 선형최적화(Linear Optimization) 방법으로 해결하기 어렵다.

이러한 무기할당 문제는 NP-complete 문제로 알려져 있으며[15], 표적과 무기의 수에 따라 계산 시간이 기하급수적으로 증가하므로 최적해를 보장하는 전수검사 알고리즘을 적용하기에 적절하지 않다. 이러한 이유로 무기할당 문제를 해결하기 위하여 최적해는 아니지만 유효한 시간 내에 해를 찾을 수 있는 다양한 메타 휴리스틱 방법에 대한 연구가 많이 이루어졌다. 대표적으로 사용된 방법들로는 개미 군단 최적화[16], 개체 군집 최적화[10], 타부 서치[7,8], 모의 담금질[3], 유전 알고리즘[4] 등이 있다. 무기할당을 위한 또 다른 접근으로는 최적해가 아닌 근사해를 찾는 방법이 있다. 무기할당 문제는 파티션 마트로이드(Matroid) 제약 아래에서 목적함수가 서브모듈라(Submodular)임이 증명되었다[17]. 마트로이드 제약 아래에서 목적함수가 서브모듈라인 경우에는 그리디 휴리스틱 알고리즘이 $(1-1/e)$의 최적성을 보장한다[1,6]. 여기서 e는 자연상수를 의미한다.

무기체계는 계층적인 의사결정 구조를 형성하고 있으므로, 의사결정 레벨에 따라 무기할당 문제의 정의가 구체화 된다. 상위 의사결정 레벨인 C2 레벨에서의 무기할당은 각 예하체계에서 교전할 표적을 할당하는 개념으로 표적-대응체계 할당의 문제로 정의된다. 실제 교전 행위를 수행하는 포대급 레벨에서의 무기할당은 시간 개념을 고려해야하므로 상위레벨보다 좀 더 복합한 형태로 발사시점을 포함한 표적-무기할당 문제로 정의된다. 특히 대 탄도탄을 비롯한 고속 표적에 대한 교전 상황에서는 표적의 속도가 매우 빠르기 때문에 교전 결심에 사용할 수 있는 시간이 매우 제한적이어서, 무기할당 시 발사 시점에 대한 고려가 필수적이다. 본 논문에서는 포대급 의사결정레벨에 적용되는 시간을 고려한 무기할당 문제(Time dependent Weapon Target Assignment : TWTA)를 다루고자한다.

TWTA 문제는 기본 WTA 문제에 비해서 복잡하므로, 표적-무기할당과 발사시점 산출을 두 단계로 나누어서 표적-무기할당이 완료된 상황에서 순차적으로 발사시점을 산출한다면 간편하게 문제를 풀 수 있다. 그러나 표적-무기할당에 따라 발사 시점 산출이 달라질 수 있으며, 교전발사시점 산출에 따라 표적-무기할당이 바뀔 수 있으므로 두 단계로 나누어진 문제가 서로 독립적이지 않다. 그러므로, TWTA 문제를 두 단계로 나누어서 해결하기 보다는 표적-무기할당-발사시점을 함께 고려하여 해를 찾는 접근이 바람직하다. D. H. Co는 $(1-1/e)$의 최적성을 보장하는 그리디 휴리스틱 알고리즘을 적용하여 표적-무기할당-발사 시점을 함께 고려하여 해결하는 방법을 제안하였다[5]. 아울러 TWTA의 문제의 목적 함수가 서브모듈러 속성을 가짐을 함께 증명하여 합리적인 방법론을 제시하였으나, 시간적 측면에서 실 무기체계에 적용할 수 있는 수준으로 처리시간을 보장하는지는 검토되지 않았다.

본 논문에서는 TWTA 문제 해결을 위한 그리디 휴리스틱 알고리즘을 제안하고자 한다. 제안 알고리즘에서는 처리 시간적 측면에서 효율적으로 구현할 수 있도록 계층적 구조를 적용한 방법을 제안하고자 하며, 제안한 방법이 실 무기체계에 적용할 수 있는 처리시간 성능을 갖는지 실험을 통해서 분석하고자 한다.

논문의 구성은 다음과 같다. 2장에서 무기할당 문제에 대한 수리적 모델을 기술하고, 3장에서는 그리디 휴리스틱을 비롯하여 처리 시간 개선을 위한 알고리즘들을 기술한다. 4장에서는 시나리오에 따른 시뮬레이션 결과 분석을 기술하며 마지막 5장에서 결론을 기술한다.

2. 무기할당 문제의 정식화

무기할당 문제를 수리적으로 모델링하기 위해서 혼합정수계획법(Mixed Linear Integer Programming)의 정식화 형식을 따른다. 우선적으로 표적-무기할당 및 발사 시점을 결정하기 위한 변수와 제약조건을 표현하기 위하여 필요한 변수들을 정의한다.

t : 표적 인덱스. $t \in \{1, ...|T|\}$
w : 발사대 인덱스. $w \in \{1, ...|W|\}$
m : 유도탄 인덱스, $m \in \{1, ..|M_w|\}$
m_w : 발사대에 적재된 유도탄 수량

FN_t : 표적 t에 할당 할 수 있는 유도탄의 개수
$ft_{w,t}$: 표적 t에 대한 발사대 w의 최초발사가능 시점
$lt_{w,t}$: 표적 t에 대한 발사대 w의 최후발사가능 시점
$PK_{w,t}$: 발사대 w에서 표적 t에 대한 요격 확률
$\theta_{w,t}(m)$: 표적 t에 발사대 w의 유도탄 m의 할당 여부, 할당 되면 0, 그렇지 않으면 1의 값을 가짐.
$\tau^l_{w,m}$: 발사대 w에서 유도탄의 m의 발사 시점
τ^{ld}_w : 발사대 상의 최소 발사 간격

본 논문에서는 위협 표적의 요격을 최대화 할 수 있는 표적 기반 무기할당 문제로 한정하고자 하여, 앞서 정의한 변수들을 활용하여 수식 (1)과 같이 목적함수를 정의하였다. 해당 목적함수는 결정변수 $\theta_{w,t}(m)$와 요격확률의 곱의 합을 최대화함으로써 요격확률이 높을 때 표적-발사대 쌍의 발사 시점에 무기할당을 하는 의미로 해석할 수 있다.

$$\max f = \sum_{t=1}^{|T|}\left[1-\prod_{w=1}^{|W|}\prod_{m=1}^{|M|}(1-PK_{w,t}\cdot\theta_{w,t}(m)))\right] \quad (1)$$

수식 (2)는 동일 유도탄이 다수의 표적에 중복하여 할당되지 않기 위한 제약 조건이다.

$$\sum_{t\in T}\theta_{w,t}(m) \le 1, \forall w\in W, m\in M_w \quad (2)$$

각 위협 표적에 할당할 수 있는 유도탄의 개수를 제한하기 위하여 수식 (3)과 같은 제약식을 적용한다.

$$\sum_{w\in W}\sum_{m\in M_w}\theta_{w,t}(m) \le FN_t, \forall t\in T \quad (3)$$

한 발사대가 연속발사를 하기 위해서는 발사절차 및 발사대 안정화를 위한 시간을 고려하여야 하므로, 수식 (4)과 같이 한 발사대 상의 연속적 발사를 위한 최소 발사 간격을 제약으로 고려한다.

$$\begin{aligned}&\left|\tau^l_w(m_i)-\tau^l_w(m_j)\right| \ge \tau^{ld}_w,\ \forall w\in W, m_i \ne m_j\\&\forall m_i,m_j\in M_w, ,\theta_{w,t}(m_i)=1,\theta_{w,t}(m_j)=1\end{aligned} \quad (4)$$

각 발사대에서 발사된 유도탄은 제한된 공간 영역 상에서 표적을 요격할 수 있다. 표적의 예상 궤적과 발사대 위치에 따른 요격 공간으로부터 교전 가능한 유도탄 발사 시간이 계산되어진다. 위협 표적에 대해서 교전 가능한 발사 시간 구간 내에서 발사 시점이 산출되어야만 교전 가능하므로, 수식 (5)와 같이 표적 t에 대한 발사대 w에서 발사 가능한 시간에 대한 제약 조건을 고려해야 한다.

$$ft_{w,t} \le \tau^l_w(m) \le lt_{w,t}, \forall m\in M_w, \theta_{w,t}(m)=1 \quad (5)$$

각 발사대는 적재된 수량만큼 유도탄을 사용할 수 있으므로, 수식 (6)과 같이 한 발사대에서 할당할 수 있는 유도탄의 개수는 적재량 이하여야 한다는 제약조건을 고려한다.

$$\sum_{t\in T}\sum_{s\in M_w}\theta_{w,t}(s) \le m_w, \forall w\in W \quad (6)$$

3. 배경 이론

3.1 그리디 알고리즘

목적함수가 아래 수식과 같은 조건을 만족할 때 서브모듈러 속성을 가진다고 할 수 있다.

$$f(A)+f(B) \ge f(A\cup B)+f(A\cap B) : A,B \subseteq V \quad (7)$$

V는 선택할 수 있는 모든 후보들의 전체 집합을 나타내고 A와 B는 그것의 부분집합이다. 목적함수의 이득이 단조로운 상승(Monotonic Increase) 이라면 그리디 알고리즘은 전체집합 V의 부분집합 중에서 제약조건을 만족하는 것 중 최적해 목적함수 값에 대해 $(1-1/e)$ 이상의 최적성을 가지는 부분집합을 찾는다[9,14]. 2장에서 기술한 본 논문의 목적함수는 [5]의 논문에서 기술된 목적함수에서 가중치 값만 제거된 형태이므로 서브모듈러 속성을 가짐을 쉽게 알 수 있다.

그리디 휴리스틱 알고리즘은 아래와 같이 의사 코드로 표현할 수 있다. 알고리즘의 기본적인 골격은 목적함수 $f: 2^V \rightarrow R_+$가 더 이상 증가하지 않거나 더 이상 선택할 수 있는 요소가 없을 때까지 반복한다. 무기할당 시 고려해야 하는 제약조건 집합을 C로 두며, 매 루프마다 C에 속하는 하나의 조건이라도 위배하는 요소는 집합 R에 담도록 한다. 그리디 휴리스틱 알고리즘의 세부 동작 과정은 매 루프마다 전체집합

V 상의 요소들에 대한 한계 이익(Marginal Gain)을 갱신하여, 한계 이익이 가장 큰 요소를 선택하는 것이다. 목적 함수 $f:2^V \rightarrow R_+$에 대해서 부분 집합 $S \subseteq V$에 대해서, 요소 $u \in V \backslash S$에 대한 한계 이익은 아래와 수식 (8)과 같이 정의된다.

$$\triangle f(u|S) := f(S \cup u) - f(S) \tag{8}$$

Algorithm 1: Greedy algorithm

Input:
$f:2^V \rightarrow R_+$
V: universal set
C: set of constraints

Output:
S: solution ($S \subseteq V$)

```
S ← ∅, R ← ∅
while V = S∪R or no improvement of f do
  for each c∈C do
    for each u∈V \ (S∪R) do
      if c(S∪u) = false then
        R ← R∪u
      end if
    end for
  end for
  u′ ← argmax_{u∈V \ (S∪R)} Δf(u|S)
  S ← S∪u′
end while
return S
```

기본적인 그리디 휴리스틱 알고리즘은 매 루프마다 argmax함수를 사용할 때 $V \backslash (S \cup R)$ 상에 있는 u에 대한 한계 이익을 계산하기 때문에 매 루프마다 $|V \backslash (S \cup R)|$번의 오라클 콜이 발생한다. 오라클 콜의 수가 처리시간이 가장 큰 영향을 주기 때문에, 오라클 콜의 수를 줄이는 방법에 대한 연구가 이루어졌다. 그 중에서 가장 기본적이면서도 효과적인 방법으로 레이지 그리디 휴리스틱(Lazy Greedy Heuristic) 방법이 있다[16,17]. 다음 장에서는 레이지 그리디 휴리스틱 알고리즘에 대해서 기술하고자 한다.

3.2 레이지 그리디 알고리즘

레이지 그리디 방법의 의사코드는 아래와 같은 의사코드로 나타낼 수 있다. 그리디 알고리즘에서 argmax를 위해서 사용되는 오라클 콜을 줄이기 위해서, 레이지 그리디 알고리즘은 모든 요소(Element)의 한계 이익을 계산하여 초기에 1회 정렬을 하고, 정렬된 결과를 매 루프마다 활용한다. 신규 할당으로 인하여 이전 단계에서 정렬된 순서 중에서 1순위의 요소의 한계 이익이 변경되는 경우에는 정렬된 순서대로 한계 이익을 재계산하여 비교하되 기존 1순위의 한계 이익보다 항상 작은 요소까지만 비교하여 정렬을 갱신한다. 목적함수가 서브모듈러 속성을 가지는 경우에는 디미니싱 리턴(Diminishing Return) 특성을 가지기 때문에 한계 이익이 증가할 수 없으므로, 앞선 정렬 결과를 활용할 수가 있다.

Algorithm 2: Lazy Greedy algorithm

Input:
$f:2^V \rightarrow R_+$
V: universal set
C: set of constraints

Output:
S: solution ($S \subseteq V$)

```
S ← ∅, R ← ∅
V ← sorted list in decreasing order by Δf(u|S)
while V = S∪R or no improvement of f do
  for each c∈C do
    for each u∈V \ (S∪R) do
      if c(S∪u) = false then
        R ← R∪u
      end if
    end for
  end for
  I ← the set of all possible elements having lager
      marginal gains than the best one in the sorted
      V\(S∪R)
  I ← sorted list in decreasing order by Δf(u|S)
  u′ ← the first element in sorted I
  update the order of elements in V\(S∪R) using
  sorted I
  S ← S∪u′
end while
return S
```

레이지 그리디 알고리즘은 전 단계의 루프에서 구축된 정렬결과를 활용할 수 있으므로, 매 회 모든 요소들에 대한 한계 이익을 계산하지 않아도 되므로 처리 시간 개선 효과를 기대할 수 있다.

본 논문에서는 레이지 그리디 방법을 기반으로 하여 그리디 알고리즘의 계산 효율성을 높이기 위해서

계층적인 구조를 적용한 새로운 방법을 제안하며 다음 장에서 자세히 기술하고자 한다.

4. 계층적 레이지 그리디 알고리즘

4.1 알고리즘 개요

본 장에서는 그리디 알고리즘의 처리 시간을 개선하기 위한 새로운 방법을 제안한다. 레이지 그리디 방법보다 시간 개선을 하기 위한 기존 연구로는 통계적 그리디, 랜덤 그리디 등의 여러 가지 방법이 존재한다. 이러한 방법들은 처리시간 성능 향상이 두드러지지만 목적함수에 대한 성능 측면에서 레이지 그리디에 미치지 못한다[2,11]. 본 장에서는 기존 성능을 보장하면서도 처리시간을 줄일 수 있는 방법을 제안하고자 한다. 제안하고자 하는 알고리즘은 전체 집합 V를 여러 개의 파티션으로 나누어 각 파티션 상에서 한계이익이 가장 큰 요소를 선택 한 뒤, 각 파티션에서 선택된 요소들 중에서 가장 한계 이익이 큰 요소를 선택하는 방식의 2단계의 계층적 구조와 레이지 그리디 알고리즘을 병합하여, 소요 시간 측면에서의 효율성을 향상시키는 방법이다. 해당 알고리즘은 기존 성능을 보장하기 때문에 $(1-1/e)$의 최적성도 보장한다.

4.2 세부 알고리즘

본 논문에서 제안하는 계층적 레이지 그리디 알고리즘은 그리디 알고리즘의 기본 골격 상에서 설계되었으며, 그리디 알고리즘과 다른 점은 한계 이익이 가장 큰 요소를 선택할 때, 전체 집합 V상에서 계산하지 않고, 이를 여러 개의 파티션으로 나누어 처리함을 통하여 시간상의 효율성을 추구한 것이다.

본 논문에서 제안하는 알고리즘에 대한 의사 코드는 아래에 기술되어 있다. 알고리즘의 입력으로 사용되는 값은 기존에 언급했던 것 이외에 전체집합 V에 대한 파티션 정보가 포함된다. P는 파티션의 인덱스에 대한 집합이다.

전체집합 V은 $|P|$개의 파티션으로 나눌 때, 아래 수식 (9)와 (10)과 같은 조건을 만족해야 하도록 나누어야 한다.

$$V = V_1 \cup V_2 \cup V_3 \cup \dots \cup V_{|P|} \tag{9}$$

$$V_i \cap V_j = \varnothing,\ \forall i,j \in P \text{ and } i \neq j \tag{10}$$

수식 (9), (10)를 만족하도록 나누어진 $|P|$개의 파티션은 하위 단계의 집합으로 하위집합 내에서 한계 이익이 가장 큰 요소를 선택하는 과정에 레이지 그리디 방식을 적용하기 위해서 의사코드의 세 번째 줄과 같이 각 파티션 별로 초기 정렬을 수행한다. 전체 집합 V에 레이지 그리디를 적용하는 것보다 하위집합에 적용하게 되면 전체 집합의 크기가 줄어들기 때문에 초기 정렬 및 부분 정렬 수행시간을 감소시킬 수 있다.

Algorithm 3: Hierarchical Lazy Greedy algorithm

Input:
$f: 2^V \to R_+$
V: universal set
C: set of constraints
P: index set of partitions
$V_1, \dots V_p$*: Partitions of the universal set*

Output:
S: solution $(S \subseteq V)$

```
1: S ← ∅, R ← ∅
2: for each p∈P do
3:   V_p ← sorted list in decreasing order by △f(u|S)
4: end
5: while V = S∪R or no improvement of f do
6:   for each c∈C do
7:     for p∈P do
8:       for each u∈V_p \ (S∪R) do
9:         if c(S∪u) = false then
10:          R ← R∪u
11:        end if
12:      end for
13:    end for
14:  end for
15:  L ← ∅
16:  for each p∈P do
17:    I_p ← the set of all possible elements having lager
          marginal gains than the best one in the sorted
          V_p\(S∪R)
18:    I_p ← sorted list in decreasing order by △f(u|S)
19:    u_p' ← the first element in sorted I_p
20:    update the order of elements in V_p\(S∪R) using
       sorted I_p
21:    L ← L∪u_p'
22:  end for
23:  u' ← argmax_{u_p'∈L} △f(u_p'|S)
24:  S ← S∪u'
25: end while
26: return S
```

의사코드 19번째 줄과 같이 각 파티션마다 레이지 그리디 알고리즘을 적용하여 한계 이익이 가장 큰 요소 $u_p{}'$를 추천한다. 의사코드 23과 같이 각 파티션의 추천 요소를 모아 그중에서 가장 한계 이익이 큰 것을 선택한다.

의사코드 17~18은 이전 루프에서 선택된 요소에 의해 한계 이익이 갱신되어 각 파티션의 정렬 구조를 재정비 하는 과정이다. 이는 기존 레이지 그리디에 사용되었던 방법을 각 파티션 별로 적용한 것이다.

모든 파티션은 수식 (9)를 만족하기 때문에, 두 단계의 과정을 거쳐서 한계 이익이 가장 큰 요소를 선택하더라도 전체집합 V 상에서 한계 이익이 가장 큰 요소를 선택한 것과 동일한 결과를 기대할 수 있다. 또한 모든 파티션은 수식 (10)을 만족하기 때문에 중복되는 연산 없이 알고리즘의 효율성을 높일 수 있다.

수식 (9)와 (10)을 만족시키면서 파티션을 나눌 수 있는 방법은 여러 가지를 생각할 수 있겠으나, 파티션을 어떻게 나누느냐에 따라서 알고리즘을 성능에 큰 영향을 줄 수 있다. 본 논문에서는 선택할 수 있는 요소들을 대응 표적 별로 파티션을 나누는 방법을 사용하였다.

일반적으로 생각하면 발사대 별로 파티션을 나누는 것이 자연스럽다고 생각할 수 있겠으나 표적 별로 파티션을 나누는 데는 큰 장점이 존재한다. 표적 별로 파티션을 나누는 것은 처리 시간 측면에서 매우 효과적인데, 그 이유는 목적함수에 있다.

수식 (1)의 목적함수를 살펴보면 무기할당 문제는 개별 표적에 여러 개의 유도탄을 할당하는 형태의 문제로 한 요소가 가지는 한계 이익의 변경이 발생하는 것은 이미 무기가 할당된 표적에 추가적으로 할당될 때이다. 그 외의 경우에는 어떤 요소이든 초기에 계산된 한계 이익이 변경되지 않는다. 만일 파티션을 표적 별로 나누게 되면, 이전 루프에서 선택된 요소가 대응하는 표적에 추가적으로 대응하는 요소들만 한계 이익을 변경하면 된다. 즉, 이전 루프에서 선택된 요소가 속해 있는 파티션 하나에 대해서만 한계 이익을 재계산하고 정렬 구조를 갱신하며 한계 이익의 변경이 없는 다른 파티션은 아무런 조치를 하지 않아도 되므로, 한계 이익과 정렬을 갱신하는데 있어서는 기존 레이지 그리디 방법에 비해서 약 $1/|P|$배의 시간을 절약 할 수 있다.

본 논문에서 제안하는 계층적 그리디 알고리즘은 파티션을 나눔으로써 초기 정렬 및 부분 정렬에 소요되는 시간을 줄일 수 있으며, 목적함수의 특수성을 잘 활용하여 대응 표적 별로 파티션을 나눔으로 인하여 획기적으로 처리 시간을 개선하였다. 본 논문에서 제안하는 알고리즘의 효율성을 확인하기 위하여 다음 장에서 실험적인 결과를 분석하고자 한다.

5. 실험 결과

본 장에서는 다수 표적에 대한 무기할당 방법으로 본 논문에서 제안하는 방법이 무기체계에 적용하기에 적합한 성능을 보이는지 분석하기 위하여 교전 시나리오 상에서 목적함수 달성도와 수행시간을 측정하였다.

목적함수 달성도 측면에서는 기본 그리디 알고리즘과 계층적 레이지 그리디 알고리즘의 성능이 달라지지 않음을 실험적으로 확인하였다. 이와 함께 처리 시간 효율성을 입증하기 위해서, 실 무기체계의 교전통제시스템이 주로 C/C++ 언어를 사용하여 개발되고 있는 상황을 고려하여, 본 논문에서 언급된 3가지 알고리즘을 Visual studio 2017상에서 C++ 언어로 구현한 뒤, i7-7500U CPU, 16GB RAM 환경 하에서 실험한 결과를 측정하였다.

시뮬레이션은 발사대를 5대로 가정하여 진행하였으며 위협 표적의 개수를 10개에서 100개까지 10개씩 증가시키며 실행하여 표적 수의 증가에 따른 알고리즘 성능을 비교하였다. 해당 알고리즘에서 표적 t에 할당 할 수 있는 유도탄의 개수(FN_t)는 1개로 가정하고 요격탄의 발사 간격은 1초로 가정하였다. 표적 t에 대한 발사대 w의 최초발사가능 시점($ft_{w,t}$)은 0에서 4초 사이의 랜덤한 값을 주었으며 표적 t에 대한 발사대 w의 최후발사가능 시점($lt_{w,t}$)은 3에서 10초 사이의 랜덤한 값을 가정하였다. 이 때 현실적인 조건을 고려하여 $ft_{w,t}$ 값과 $lt_{w,t}$ 값 사이는 3초 이상의 간격을 갖도록 설정하였다. 또한 표적 t에 발사대 w를 할당할 때의 보상($PK_{w,t}$: 요격확률)은 0에서 99 사이의 랜덤한 값으로 가정하였다.

계층적 레이지 그리디 방법은 기본 그리디 방법의 기본 프레임 내에서 동작하고 오라클 콜을 줄여서 계산 시간만 개선한 방법이기 때문에 목적함수 달성도 측면에서는 그 성능이 달라지지 않으며, 실험적인 결과를 통해서 분명히 확인 할 수 있다.

Table 1. Objective value and performance of greedy algorithm

Number of targets	Greedy	Lazy Greedy	Hierarchical Lazy-Greedy
10	761.25	761.25	761.25
20	1569.51	1569.51	1569.51
30	2319.36	2319.36	2319.36
40	2941.82	2941.82	2941.82
50	3286.20	3286.20	3286.20
60	3520.40	3520.40	3520.40
70	3694.57	3694.57	3694.57
80	3829.13	3829.13	3829.13
90	3882.83	3882.83	3882.83
100	3962.09	3962.09	3962.09

본 그리디 알고리즘에 대한 레이지 그리디와 계층적 레이지 그리디 알고리즘의 처리 시간 비교 결과는 Fig. 1과 같이 나타났다.

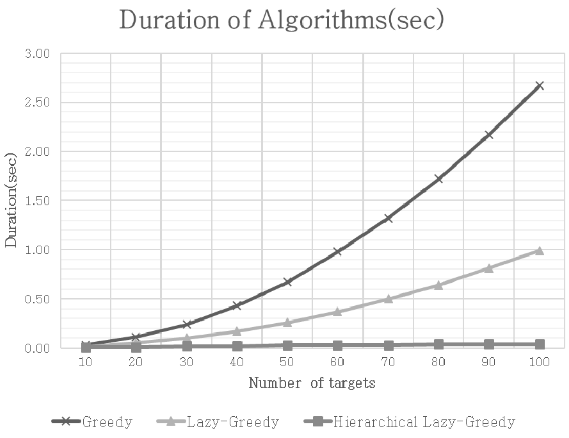

Fig. 1. Duration of algorithms

표적의 개수가 10개에서 100개까지 증가할 때 처리 시간도 함께 증가하지만 그리디 알고리즘에 비해서는 레이지 그리디 알고리즘의 증가폭이 적으며, 레이지 그리디 알고리즘에 비해서는 계층적 레이지 그리디 알고리즘 처리 시간이 적음을 볼 수 있다. 표적의 개수가 10개일 경우 알고리즘의 처리시간은 그리디 알고리즘이 0.03초, 계층적 레이지 그리디 알고리즘이 0.01초로 큰 차이가 나지 않지만 표적의 개수가 100개까지 증가할 경우 그리디 알고리즘이 2.67초, 레이지 그리디 알고리즘이 0.99초, 계층적 레이지 그리디 알고리즘이 0.04초로 처리 시간이 순차적으로 감소함을 확인할 수 있다. 또한 그리디 알고리즘과 계층적 레이지 그리디 알고리즘 사이의 처리 시간 차이가 약 67배정도로 나타난다. 해당 시뮬레이션에서 교전해야하는 표적수가 10개에서 100개로 증가하는 동안 계층적 레이지 그리디 알고리즘의 처리 시간이 0.01초에서 0.04초로 소폭 증가하며 그리디 알고리즘에 비해 짧은 처리 시간을 나타냈다.

일반적으로 무기할당 알고리즘에서 동시에 처리해야하는 표적수가 증가할수록 처리 시간에 대한 차이가 더 커질 것으로 예상할 수 있다. 다수의 고속비행 표적에 대해 무기할당 시, 그리디 알고리즘의 경우 표적 수가 100개 일 때 처리 시간이 3초에 가까워지므로 실 무기체계에 적용 시 무기할당 처리에 대한 지연시간으로 인한 부담을 갖게 된다. 그러나, 본 논문에서 제안하는 계층적 레이지 그리디 알고리즘은 표적수가 늘어남에 따라 소요시간의 증가폭이 매우 적으며 실 처리시간도 표적 100개에 대해 0.04초로 매우 작으므로, 기존 알고리즘과 동일한 성능을 보이면서도 동시교전 표적 수 증가에 따른 소요 시간은 현저히 감소시켰다.

6. 결 론

NP-complete 문제로 알려진 무기할당 문제는 문제의 복잡성으로 인해 풀이가 어렵기도 하지만 촌각을 다투는 전장상황에서 의사결정을 지원해야하므로 처리 시간에 대한 제약이 매우 엄격한 문제이다. 무기할당 문제를 다루는 수많은 논문이 있으나 엄격한 처리시간 제약 아래에서 실 체계의 적용성에 대한 고찰은 매우 부족하였다. 본 논문에서는 기존의 무기할당 방법 중에서 $(1-1/e)$의 근사해를 보장하는 그리디 휴리스틱 알고리즘의 처리시간을 개선할 수 있는 계층적 레이지 그리디 알고리즘은 제안하여 실험적으로 그 성능을 입증하였으며, 실험 결과 분석을 통하여 실 무기체계에 대한 적용성이 뛰어남을 확인하였다. 향후에는 무기할당의 처리시간은 효율적으로 유지하면서도 목적함수 달성도 측면에서의 성능개선의 연구가 지속

되어야 하고 실질적인 요격확률을 도입한 시뮬레이션을 진행할 예정이다.

References

[1] A. Krause and D. Golovin, "Submodular Function Maximization," Tractabilikty: Practical Approaches to Hard Problems, Vol. 3, p. 19, 2012.

[2] Badanidiyuru, A., and Vondrak, J. "Fast Algorithms for Maximizing Submodular Functions," In SODA, pp. 1497-1514, 2014.

[3] C. D. G. S. Kirkpatrick and M. P. Vecchi, "Optimization by Simulated Annealing," Science, Vol. 220, No. 4598, pp. 671-680, May 1983.

[4] D. e. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning," Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1989.

[5] D. H. Co and H. L. Choi, "Greedy Maximization for Asset-Based Weapon-Target Assignment with Tim-Dependent Reward," Cooperative Control of Multi-Agent Systems Theory and Applications, First Edition, pp. 115-139, 2017.

[6] Edmonds, J., "Matroids and the Greedy Algorithm," Mathematical Programming, 1971.

[7] F. Glover, "Tabu Sesarch - Part i," ORSA Journal on Computing, Vol. 2, No. 3, pp. 190-206, Summer, 1989.

[8] F. Glover, "Tabu Sesarch - Part ii," ORSA Journal on Computing, Vol. 2, No. 1, pp. 4-32, Winter 1990.

[9] Fujishige, A., "Submodular Functions and Optimization," Elsevier Science, 2nd edition, 2005.

[10] J. Kennedy and R, Eberhart, "Particle Swarm Optimization," Vol. 4, Neural Networks, Perth, WA: IEEE, pp. 1942-1948, Dec. 1995.

[11] Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrak, J. and Krause, A., "Lazier Than Lazy Greedy," In AAAI, 2015.

[12] Minoux, M. "Accelerated Greedy Algorithms for Maximizing Submodular Set Functions," Optimization Techniques, LNCS 234-243, 1978a.

[13] Minoux, M. "Accelerated Greedy Algorithms for Maximizaing Submodular Set Functions," In Proc. of the 8th IFIP Conference on Optimization Techniques. Springer. 1978a.

[14] Nemhauser, G. L., Wolsey, L. A. and Fisher, M. L. "An Analysis of Approximations for Maximizing Submodular Set Functions," - I. Mathematical Programming, 1978.

[15] S. P. Lloyd and H. S. Witsenhausen, "Weapon Allocation is Np-Complete," Summer Computer Simulation Conference, 1986.

[16] V. M. M. Dorigo and A. Colorni, "Ant System: Anautocatalytic Optimizing Process," Dipartimento di Elettronica e Informazione Politecnico di milano, Piazza Leonardo da Vinci 32 20133 Milano, Italy, Tech. Rep., 1991.

[17] Y. L. Zengfu Wang, Xuezhi wang and Q. Pan, "Weapon Target Assignment Leveraging Strong Submodularity," in Proceeding of the IEEE International Conference on Information and Automation, Yinchuan, China, pp. 74-79, August 2013.

[18] Hyesun Jeong, Jieun Kim, Hyeseung Koh and Ohkyun Jeong, "Weapon-Target Assignment Algorithm using Submodular Function Maximization for Multi-Target Engagement," KIMST Annual Conference Proceedings, Vol. 2018, No. Autumn, pp. 881-882, 2018.