한국군사과학기술학회지 제13권 제1호, pp. 1~8, 2010년 2월

학술논문 유도무기 부문

Munkres 최적할당 기법을 적용한 무기할당 알고리즘

A Weapon Assignment Algorithm Using the Munkres Optimal Assignment Method

김 지 은* 신 진 화* 조 길 석*
Ji Eun Kim Jin Hwa Shin Kil Seok Cho

Abstract

This paper presents global and optimal solution for weapon assignment problems using the Munkres assignment algorithm. We propose a new modeling method of weapon assignment problems concerning some constraints of weapon systems. In this paper, we compares the Munkres weapon assignment algorithm with two other algorithms employing a search tree model in terms of computational complexity and performance. One is an optimal algorithm using exhausted search and the other is a greedy algorithm which selects the first search result as a solution. The experiment results show that the Munkres weapon assignment algorithm has better performance and less computational complexity in comparison with the two other algorithms.

Keywords : Weapon Assignment(무기할당), Munkres Assignment Algorithm(Munkres 할당 알고리즘), Search Tree(탐색 트리)

1. 서 론

다 표적 동시교전 임무를 효과적으로 수행하기 위해 각 표적에 대하여 최적의 무기를 할당하는 문제가 연구되고 있다. 최적 무기할당이라는 말의 의미는 두 가지 측면에서 해석할 수 있다. 첫 번째는 각 표적을 독립적인 대상으로 간주하여 각 표적에 최적의 무기를 할당하고 먼저 교전에 들어가는 표적이 최적의 무기를 선점하는 것이다. 두 번째는 하나의 표적에 무기를 할당한 것이 다른 표적의 무기할당에 미치는 영향을 고려하여 수행될 모든 교전에 대해서 총체적으로 가장 효율적인 무기할당을 수행하는 것이다. 본 논문은 후자의 관점에서 최적의 무기할당 문제를 해석하여 접근하고자 한다.

최적의 무기할당 문제는 1986년도에 Lloyd에 의해 NP-Complete 문제에 해당한다고 분석되었다[1]. 그 동안 무기할당 문제를 해결하기 위한 방법으로 인공지능 기법들에 기반을 둔 다양한 학습(Heuristic) 알고리즘들이 제안되었다[2~7]. 일반적으로 자원할당 문제는 모두 NP-Complete 문제로 분류되고 있으나, 실제적으로 Munkres 알고리즘은 cost 행렬을 이용하여 polynomial time 내에 최적의 자원할당 결과를 찾아낸다[8,9]. 그래서 본 논문에서는 복잡한 학습 방법을 이용하지 않고, 단순하고 빠른 Munkres 알고리즘을 적용할 수 있도록

† 2009년 9월 7일 접수~2009년 11월 20일 게재승인
* 국방과학연구소(ADD)
책임저자 : 김지은(newjieun@naver.com)

무기할당 문제를 새롭게 모델링 하였다. 또한 본 논문에서 제안한 모델은 실제 무기체계에서 필요한 몇 가지 제약조건을 수용할 수 있도록 하였다.

본 논문에서 제안한 알고리즘의 성능을 확인하기 위해서 다른 두 알고리즘과 비교하였다. 이들 알고리즘은 무기할당 문제를 탐색 트리로 모델링한 것으로 하나는 Exhausted Search 기법을 이용한 최적 알고리즘이고, 다른 하나는 DFS(Depth First Search)기법을 이용한 Greedy 알고리즘이다. 본 논문에서는 Munkres 알고리즘을 적용한 무기할당 알고리즘을 MWA(Munkres Weapon Assignment)라고 하고, 최적 트리 탐색 알고리즘을 적용한 무기할당 알고리즘을 OWA(Optimal Weapon Assignment)라고 하며, Greedy 트리 탐색 알고리즘을 적용한 무기할당 알고리즘을 GWA(Greedy Weapon Assignment)라고 부른다.

본 논문의 2장에서는 Munkres 알고리즘을 소개하고 무기할당 문제를 어떻게 Munkres 알고리즘으로 모델링 하였는지를 기술한다. 3장에서는 무기할당 문제를 탐색 트리에 어떻게 모델링 하였는지를 보여주고, OWA 알고리즘에 대하여 살펴본다. 4장에서는 GWA 알고리즘에 대해서 알아보고, 5장에서는 세 가지 알고리즘에 대해 시뮬레이션을 수행한 결과를 비교분석하여 제안한 알고리즘을 검증한다. 마지막으로 6장에서 연구결과를 요약하며 향후 연구방향을 제시한다.

2. Munkres 알고리즘을 적용한 무기할당

본 장에서는 Munkres 알고리즘에 대하여 살펴보고 무기할당 문제에 적용할 수 있도록 새롭게 모델링한다.

가. Munkres 알고리즘

Munkres 알고리즘은 polynomial time 내에 최적의 자원할당을 할 수 있는 알고리즘으로 bipartite 그래프 모델링에 기반을 둔다. 이 알고리즘은 Harold Kuhn에 의해 “Hungarian method”로 소개되었고[8], 이후 Munkres에 의해서 Harold Kuhn의 알고리즘보다 더 빠른 polynomial time 내에 수행할 수 있는 알고리즘으로 개선되었다[9]. Munkres에 의해 제안된 알고리즘의 시간 복잡도는 $n \times n$ cost 행렬에 대해서 $O(n^4)$이며, 이후 Edmonds와 Karp에 의해서 $O(n^3)$으로 개선되었다.

Munkres 알고리즘은 음수가 아닌 cost 행렬을 정의하고, cost의 합이 최소가 되는 할당 결과를 찾는다. Munkres 알고리즘에 사용되는 용어들을 정의하면 아래와 같다.

- worker : 자원을 의미한다.
- job : 자원을 할당받는 일을 의미한다.
- cost : 자원을 일에 할당할 때 드는 비용을 의미한다.
- cost 행렬 : 자원을 일에 할당할 때 각각의 쌍이 가지는 비용을 하나의 행렬로 구성한 것이다. Table 1은 3개 worker를 3개 job에 할당하는 문제에 대한 cost 행렬 예를 보여준다. 여기서 worker1을 job1에 할당하는데 드는 비용은 3이다.

Table 1. Cost 행렬 예

	Job1	Job2	Job2
Worker1	3	4	2
Worker2	5	3	1
Worker3	3	2	4

- $*zero$: Cost 행렬에서 원소 값이 0이고 해당 행과 열에 $*zero$가 없는 원소를 $*zero$로 정의한다.
- $'zero$: $uncovered$인 행과 열에 있는 0을 의미한다.
- $covered$: $*zero$나 $'zero$가 있는 열이나 행을 $covered$로 정의한다.
- $uncovered$: $covered$로 되지 않은 행과 열을 의미한다.
- C : $n \times m$ cost 행렬 이다. 여기서 n은 worker의 수이고 m은 job의 수이며, k는 n이나 m 중에서 최소 값으로 정의한다.
- Z-리스트 : Cost 행렬에서 $'zero$와 $*zero$를 순서대로 저장한 배열이다.

위에 정의된 기호를 이용한 Munkres 알고리즘의 할당 절차는 아래 step1부터 step7까지에서 보여준다.

- **Step0** : Job과 worker가 일대일로 매치 되는 cost 행렬 C를 생성한다.
- **Step1** : C의 각 행에서 가장 작은 수를 찾아 각 행의 원소들로부터 뺀다.
- **Step2** : C에서 0을 찾는다. 0이 있는 행과 열에 $*zero$가 없다면 이를 $*zero$로 설정한다.
- **Step3** : $*zero$가 있는 열을 모두 $covered$로 설정한

후 k열이 모두 $covered$로 설정되었다면 step7을 수행하고 그렇지 않으면 step4를 수행한다.

- **Step4** : C에서 $uncovered$ zero를 찾아서 $'zero$로 설정한다. $'zero$가 있는 행에 $*zero$가없으면 step5를 수행한다. 그렇지 않으면 해당 행을 $covered$로 설정하고, $*zero$가 있는 열을 $uncovered$로 설정한다. $uncovered$ 0이 없어질 때까지 반복한 후 step6을 수행한다.
- **Step5** : C에 대한 Z-리스트를 생성한다. step4에서 찾은 $'zero$는 Z_0로 표시하고, Z_0의 열에 있는 $*zero$는 Z_1으로 표시하며, Z_1의 행에 있는 $'zero$는 Z_2로 표시한다. 해당하는 열에 $*zero$가 없는 $'zero$가 나타나면 Z-리스트가 완성된다. Z-리스트에 있는 $*zero$는 C에서 *표시를 지우고, Z-리스트에 있는 $'zero$는 C에서 모두 $*zero$로 설정한다. C에서 모든 $'zero$에 대해서 $'$표시를 지우고 모든 열과 행을 $uncovered$로 설정한 후 step3로 돌아간다.
- **Step6** : Step4에서 찾은 값을 모든 $covered$ 행의 값으로 더하고, 모든 $uncovered$ 열의 값으로 뺀 후 step4로 돌아간다.
- **Step7** : C에서 $*zero$로 선택된 원소들이 나타내는 job과 worker의 쌍이 최적의 할당 결과이다.

나. Munkres 알고리즘을 적용한 무기할당 모델링

본 장에서는 Munkres 알고리즘을 적용하여 무기할당 문제를 아래와 같이 모델링한다.

1) 무기할당 모델링

Munkres 알고리즘을 무기할당 문제에 적용하기 위해서 정의해야 할 중요한 사항은 cost 행렬이다. 본 논문에서는 cost 행렬을 격추확률을 기반으로 만든다. 대공무기체계의 격추확률에 대한 예는 Table 2에서 보여준다. 격추확률은 일반적으로 표적 거리와 유도탄의 성능에 의해서 결정되는 것으로 최소한의 무기를 이용하여 최다 표적을 격추시키기 위해서는 격추확률이 높은 무기를 추천하여 교전의 실패가 없도록 하는 것이 유리하다. 이미 Li Hongrui에 의해서 격추확률을 기반으로 한 무기할당 알고리즘이 제안된 바 있다[7].

Munkres 알고리즘은 음수가 아닌 cost 행렬을 사용하며 cost의 합이 최소가 되는 할당 결과를 만든다. 격추확률을 이용한 cost 행렬을 기반으로 Munkres 알고리즘을 무기할당 문제에 적용하려면 아래와 같은 정의가 필요하다.

- worker : 무기체계
- job : 위협표적
- cost : 100/격추확률
- n : 무기체계 수
- m : 위협표적 수
- C : $n \times m$ cost 행렬

Table 2. 대공무기체계의 격추확률 테이블

	40km	50km	60km
Worker1	95	78	51
Worker2	78	78	70
Worker3	88	50	40

Table 3. 격추확률을 이용한 cost 행렬

	Job1	Job2	Job2
Worker1	1.05	128	1.96
Worker2	1.28	1.28	1.42
Worker3	1.13	2.0	2.5

Cost는 자원할당에 드는 비용이므로 표적에 무기를 할당했을 때 비용을 최소로 하려면, cost는 격추확률에 반비례하는 값으로 설정하여야 하며 Table 2를 이용한 cost 행렬은 Table 3과 같다.

2) 제약조건을 고려한 무기할당 모델링

Munkres 알고리즘에 입력되는 cost 행렬을 위와 같이 모델링하면 Munkres 알고리즘을 변경하지 않고 그대로 활용할 수 있다. Munkres 알고리즘은 어떤 할당 문제이든 cost 행렬만 만들어주면 되기 때문에 다음의 추가적인 제약조건들을 쉽게 수용할 수 있다.

- **조건 A** : 무기체계의 성능을 반영하기 위해서 무기체계별 동시교전 표적 수가 제한된다.
- **조건 B** : 위협이 매우 높다고 판단된 표적에 대해서 이중무기 할당을 고려한다.

조건 A를 고려한 모델은 worker에 상응하는 무기체계를 동시교전 가능한 표적 수만큼 반복하여 아래와

같이 정의하면 된다.

- n : 무기체계 × 동시교전 표적 수
- m : 위협표적 수

조건 B를 고려한 모델은 job에 상응하는 개념으로 고 위협으로 판단되는 표적은 두 번 반복하여 아래와 같이 정의하면 된다.

- n : 무기체계 수
- m : 위협표적 수 + 고 위협으로 판단된 표적 수

조건 A와 조건 B를 모두 적용한 모델은 조건 A와 조건 B를 모두 고려하면 job과 worker의 수는 아래와 같이 정의하면 된다.

- n : 무기체계 × 동시교전 표적 수
- m : 위협표적 수 + 고 위협으로 판단된 표적 수

조건 A와 B를 적용한 모델은 worker와 job의 수가 모두 늘어났기 때문에 제약조건을 고려할 때 시간적인 효율이 떨어질 것이라고 생각하기 쉽다. 그러나 Munkres 알고리즘은 polynomial time을 보장하여 주기 때문에 여기에 상수 배의 시간이 늘어난다고 하더라도 여전히 polynomial time을 보장해 준다. Munkres 알고리즘의 시간 복잡도가 $n \times n$ cost 행렬에 대해서 $O(n^4)$인 것을 고려하면 MWA 알고리즘의 시간 복잡도는 $O(m^4)$가 된다.

3. 최적 트리 탐색 알고리즘을 적용한 무기할당

2장에서 bipartite graph에 기반을 둔 Munkres 알고리즘으로 무기할당 문제를 모델링한 반면에 본 장에서는 탐색 트리를 기반으로 무기할당 문제를 모델링하여 계산하는 방법을 적용하고자 한다.

탐색 트리는 정보를 노드와 가지로 이루어진 트리 형태로 데이터를 저장하는 구조이다. 탐색 트리의 노드와 가지는 저장되는 순서에 따라 의미를 가지며, 어떠한 문제를 탐색 트리로 모델링하기 위해서는 트리의 구성요소인 노드, 레벨, 가지 및 경로와 주어진 문제의 요소들과의 연관성을 찾아야 한다. 무기할당 문제의 경우에는 아래와 같이 그 연관성을 정의할 수 있다.

- 노드 : 위협표적에 할당되는 무기체계
- 레벨 : 위협표적 수
- 가지 : 위협표적에 할당되는 무기체계의 cost
- 경로 : 무기할당 결과
- 최적 무기할당 결과 : 모든 가지의 합이 가장 최소가 되는 경로

무기할당 문제를 탐색 트리로 모델링하기 위한 기본적인 정의들은 아래와 같다.

- n : 무기체계 수
- m : 위협표적 수
- L_i : 트리의 i번째 레벨이다. 여기서 i는 $\{0 \le i \le m\}$ 범위이다.
- N_{ij} : 트리 레벨 L_i의 j번째 노드이다. 여기서 i와 j는 각각 $\{0 \le i \le m\}$, $\{0 \le j \le n^m - 1\}$이고, N_{00}는 루트 노드를 의미한다.
- B_{ij} : 트리 레벨 L_{i-1}의 노드와 트리 레벨 L_i의 노드를 연결하는 가지이다. 여기서 i와 j는 각각 $\{0 \le i \le m\}$, $\{0 \le j \le n^m - 1\}$의 범위이다.
- C_{ij} : B_{ij}의 cost 값이다. cost는 '100/격추확률' 값으로 정의한다.
- P_{ij} : 루트 노드 N_{00}에서 노드 N_{ij}까지 경로에 속한 노드들의 집합이다. 여기서 i와 j는 각각 $\{0 \le i \le m\}$, $\{0 \le j \le n^m - 1\}$의 범위이다.
- P_{ms} : 최적 무기할당 결과이며, 여기서 s는 $\{0 \le s \le n^m - 1\}$ 범위이다.
- C_{ij} : 경로 P_{ij}에 있는 가지들의 cost 합이며, 아래 수식 (1)과 같이 나타낼 수 있다.

$$C_{ij} = \sum_{k=1}^{i} c_{kx_k} \tag{1}$$

$$\begin{cases} k = i, & x_k = j \\ 1 \le k < i, & x_k = \dfrac{x_{k+1}}{n} + 1 \end{cases}$$

무기할당을 수식 (1)과 같이 모델링한 후 아래 수식 (2)와 같이 cost의 합이 최소가 되는 경로 P_{ms}를 찾으면 이 경로가 최적 무기할당 결과가 된다. P_{ms}를 찾기

위해서는 $\{0 \le j \le n^m - 1\}$ 범위에 속하는 모든 j에 대해서 C_{mj}를 계산해야 하므로 시간 복잡도는 $O(n^m)$이 된다.

$$C_{ms} = MIN(C_{m1,\ldots,}C_{m(n^m)}) \quad (2)$$

탐색 트리를 이용한 무기할당 모델의 정의에 따라서 트리를 그리면 Fig. 1과 같다. Fig. 1은 트리를 구성하고 있는 요소인 트리의 레벨, 노드, 가지 및 경로를 각각 표현하고 있다. 트리 레벨 L_0는 회색으로 표시된 루트 노드 N_{00}만을 가진다. 루트 노드는 위협표적과 무기체계에 연관되지 않으며 트리 구조를 나타내기 위해 필요한 가상의 노드이다. 트리의 레벨은 L_0을 제외하고 위협표적의 개수만큼 생긴다. 일반적인 노드들은 노란색으로 표시 했으며 리프 노드는 주황색으로 표시하였다. 초록색으로 표시한 노드들은 하나의 경로에 속하는 노드이며, 화살표가 노드와 노드를 잇는 가지를 나타내며, 각 가지는 cost 값을 가진다. 각 리프 노드에서 루트 노드까지의 경로가 각각의 무기할당 결과이다. 그 중에서 최소의 cost를 가지는 경로인 P_{ms}가 탐색 트리 알고리즘이 찾는 무기할당 결과이다.

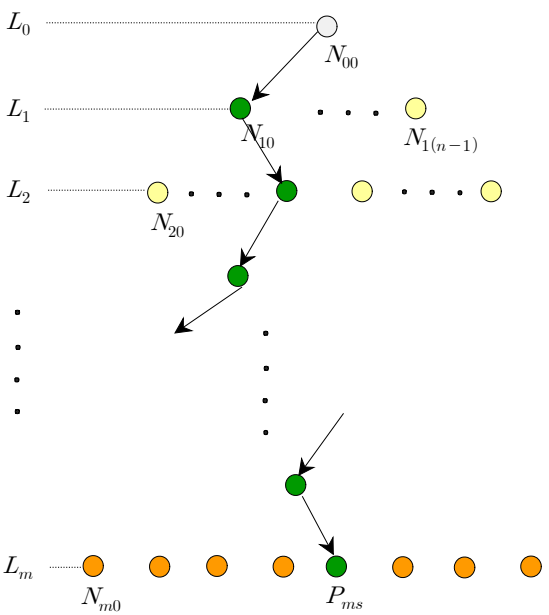

Fig. 1. 탐색 트리를 이용한 무기할당 모델

최적 트리 탐색 알고리즘을 이용한 무기할당 모델은 Munkres 알고리즘 적용한 모델과 마찬가지로 제약조건들을 고려할 수 있으나 그 방법에는 차이가 있다. Munkres 알고리즘에서 worker와 job의 수를 조정하고, cost 행렬에 반영시킨 것처럼 이 알고리즘에서도 각 레벨의 노드 수를 조정하는 방법을 사용한다면 수행시간은 기하급수적으로 증가하게 되므로 동일한 방법을 적용할 수가 없다. 그래서 이 알고리즘은 트리를 탐색하는 과정 중에 각 노드를 탐색할 때마다 현재까지 이루어진 무기할당 결과가 제약조건을 만족하는지 여부를 확인하는 방식을 사용한다.

4. Greedy 트리 탐색 알고리즘을 적용한 무기할당

Greedy 알고리즘도 3장에서와 동일하게 탐색 트리를 적용하여 무기할당 문제를 모델링한다. Greedy 알고리즘은 최적의 할당 결과를 필요로 하지 않기 때문에 트리 탐색 중에 얻어지는 첫 경로를 할당결과로 한다. 이 때 사용하는 탐색 알고리즘은 Fig. 2와 같이 depth first search가 적절하다. Fig. 2의 트리에서 각 노드에 나타난 숫자와 화살표는 depth first search가 노드를 탐색하는 순서를 나타낸 것이다.

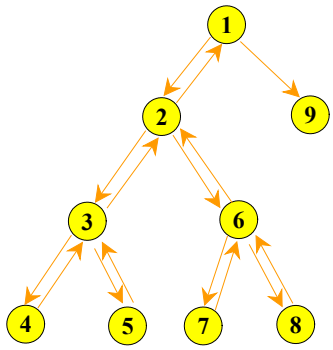

Fig. 2. Depth first search

Greedy 알고리즘에 제약조건을 적용하는 방법도 3장의 알고리즘과 동일하게 트리를 탐색하는 과정 중에 각 노드를 탐색할 때마다 제약조건을 만족하는지를 확인한다. 조건을 만족하는 경로가 처음으로 나타나면 그것을 할당 결과로 한다. 이 알고리즘은 OWA 알고리즘보다 시간이 적게 걸리지만 제약조건을 고려하는 경우에는 첫 번째 경로에서 해답을 얻을 수 없

는 경우가 많으며, 최악의 경우에는 OWA 알고리즘과 동일하게 $O(n^m)$의 시간을 소모하게 된다.

5. 시뮬레이션 결과분석

이 장에서는 본 논문에서 제안한 알고리즘의 성능을 시뮬레이션을 통해서 검증하고자 한다. 시뮬레이션에 사용하는 무기체계의 종류는 3가지이며 표적 수는 10개에서 50개까지 가변적으로 적용하여 무기체계의 수에 따라 시뮬레이션을 수행하였다. 총 세 가지 경우에 대하여 시뮬레이션을 수행하였고, 각 시뮬레이션의 제약조건은 Table 4와 같다. 시뮬레이션1은 가상적인 상황으로 각 무기체계의 수와 동시교전 표적 수에 제한이 없는 경우이고, 시뮬레이션2는 무기체계별 동시교전 수를 제한시킨 것이며, 시뮬레이션3은 시뮬레이션2에 이중무기 할당 조건을 추가한 것이다.

Table 4. 시뮬레이션별 제약조건

시뮬레이션	내용
1	· 조건 A와 조건 B를 모두 무시한다. · 무기체계의 수는 제한이 없다. · 무기체계가 동시교전 할 수 있는 표적의 수는 제한이 없다.
2	· 조건 A를 고려한다.
3	· 조건 A와 조건 B를 모두 고려한다.

Table 5. 시뮬레이션을 위한 격추확률

표적거리 (km) / 무기체계 종류	20	30	40	50	60	70	80	90	100
무기체계 X	60	80	90	95	90	80	50	30	10
무기체계 Y	40	60	80	90	95	90	70	40	10
무기체계 Z	80	90	95	90	70	60	20	10	5

각 시뮬레이션의 표적 집합은 주어진 표적 수에 따라 100km내에서 무작위로 생성하였으며, Table 5의 격추확률을 이용하여 생성된 cost 행렬을 이용하여 각 알고리즘의 cost를 계산하였다. OWA 알고리즘은 무기체계의 종류가 3개이고 표적 수가 30개 이상이면 펜티엄 3.4GHz PC 환경에서 수행 시간이 2년 이상 걸리므로 표적 수가 25개 이하인 것에 대해서만 시뮬레이션을 수행하였다. 시뮬레이션 결과를 분석하기 위한 성능인자로는 각 알고리즘의 할당결과가 갖는 cost의 합을 사용하였다. 알고리즘의 성능은 cost의 합이 적을수록 좋은 것으로 판단할 수 있다. 왜냐하면 cost의 합이 적다는 것은 할당된 무기체계가 해당 표적을 격추시킬 확률이 높은 것을 의미하므로 cost의 합이 적을수록 격추 성공률이 높기 때문이다.

아무런 제약조건 없이 가상적인 환경을 설정한 시뮬레이션1에 대한 결과는 아래 Table 6과 같다. MWA 알고리즘과 OWA 알고리즘의 성능이 동일한 것으로 보아 MWA 알고리즘도 최적의 성능을 나타내고 있다. 반면에 GWA 알고리즘의 성능은 두 알고리즘에 비해 떨어지며 표적 수가 많아질수록 성능 차이가 더 커지는 경향을 보인다.

Table 6. 시뮬레이션1의 결과

표적 수	무기체계 수			MWA	OWA	GWA
	X	Y	Z			
10	1	1	1	31.6	31.6	34.0
15	1	1	1	49.1	49.1	53.8
20	1	1	1	65.0	65.0	71.1
25	1	1	1	71.1	71.1	78.8
30	1	1	1	87.2	-	96.6
40	1	1	1	108.9	-	120.9
50	1	1	1	121.7	-	136.6

시뮬레이션2는 조건 A를 고려한 것으로 조건 A에서 제한하는 동시교전 표적 수는 Table 7과 같이 각 무기체계마다 다르게 설정하였다. Table 8에서 무기체계 종류별 무기체계의 수는 최소한의 무기체계로 주어진 표적을 모두 교전할 수 있도록 설정하여 각 알고리즘의 성능을 확인하고자 하였다. Table 8의 결과를 살펴보면 MWA 알고리즘의 성능은 OWA 알고리즘과 동일하다. 이것은 MWA 알고리즘은 조건 A를 반영하는 것과 관계없이 최적의 성능을 낸다는 것을 의미한다. 반면에 GWA 알고리즘은 두 알고리즘과의

성능 차이가 시뮬레이션1에서 보다 더 나는 것으로 보아 제약조건 A를 추가하면 성능이 더 떨어지는 것을 알 수 있다.

Table 7. 무기체계별 동시교전 표적 수

	무기체계 X	무기체계 Y	무기체계 Z
동시교전 표적 수	3	4	5

Table 8. 시뮬레이션2의 결과

표적 수	무기체계 수			동시교전 가능한 총 표적 수	MWA	OWA	GWA
	X	Y	Z				
10	1	1	1	12	31.6	31.6	40.7
15	2	1	1	15	49.9	49.9	80.3
20	2	2	2	24	65.1	65.1	93.3
25	4	1	2	26	73.7	73.7	95.7
30	4	2	2	30	89.0	-	117.7
40	3	4	3	40	109.3	-	159.3
50	3	4	5	53	123.6	-	186.8

시뮬레이션3은 시뮬레이션2에 조건 B를 추가한 것으로 이중무기를 할당하는 표적 수는 Table 9와 같이 주어진 표적 수에 따라 다르게 설정하였다. Table 10의 무기체계 종류별 무기체계의 수는 시뮬레이션2와 동일한 개념으로 주어진 표적과 모두 교전할 수 있으면서 무기체계 수를 최소한으로 사용하도록 설정하여 각 알고리즘의 성능을 확인하였다. Table 10의 결과를 보면 MWA 알고리즘의 결과와 OWA 알고리즘의 결과는 같다. 이는 MWA 알고리즘은 제약조건을 추가하여도 여전히 최적의 성능을 보이는 것을 의미한다. 반면에 GWA 알고리즘은 두 알고리즘에 비해 성능이 현격히 떨어지는 것을 볼 수 있다.

Table 9. 표적 수에 따른 이중무기할당 표적 수

표적 수	10	15	20	25	30	40	50
이중무기 할당 표적 수	2	4	6	8	10	12	14

Table 10. 시뮬레이션3의 결과

표적 수	무기체계 수			동시교전 가능한 총 표적 수	MWA	OWA	GWA
	X	Y	Z				
10	1	1	1	12	36.6	36.6	65.3
15	2	2	1	19	55.4	55.4	74.3
20	4	1	2	26	68.3	68.3	82.6
25	5	2	2	33	83.3	83.3	94.4
30	3	4	3	40	91.2	-	115.3
40	4	4	5	53	111.9	-	137.6
50	5	6	5	64	122.6	-	161.5

지금까지 3가지 경우에 대한 시뮬레이션의 수행 결과를 살펴보았다. OWA 알고리즘은 최적 알고리즘으로 최적의 성능을 보여주었으나 시간적인 측면에서는 m개의 표적과 n개의 무기체계에 대해서 $O(n^m)$의 시간 복잡도를 가지므로 실 무기체계에 적용하기에는 적합하지 않음을 알 수 있다. 그리고 GWA 알고리즘은 다른 두 알고리즘에 비해 성능이 현저히 떨어지며 제약조건을 추가하면 성능이 더 떨어지므로 제약조건이 다양한 실 무기체계에 적용하기에 적합하지 않다. 또한 시간 복잡도도 최악의 경우에 $O(n^m)$과 같아지므로 수행 시간의 예측이 어려운 알고리즘이다. 반면에 MWA알고리즘은 최적 알고리즘인 OWA 알고리즘과 성능이 동일하고, 시간 복잡도도 $O(m^4)$으로 효율적이므로 실 무기체계에 적용하기에 적합한 알고리즘이라는 것을 알 수 있다.

6. 결 론

본 논문에서는 다 표적 동시교전 환경에서 최소의 무기를 사용하여 최다 표적을 격추시킬 수 있도록 하기 위해 효율적인 무기할당 알고리즘을 제안하였다. 최다 표적과 교전하기 위해서는 하나의 표적에 대한 무기할당 결과가 다른 표적의 무기할당에 미치는 영향을 고려한 총체적인 판단이 필요하다. 본 논문에서는 자원할당을 위한 알고리즘의 하나인 Munkres 알고리즘으로 무기할당 문제를 모델링하였다. Munkres 알고리즘은 cost 행렬을 사용하여 최소 cost가 되는 할당

결과를 찾는 알고리즘이다. 또한 Munkres 알고리즘을 적용한 무기할당 알고리즘을 확장하여 두 가지의 제약조건을 추가로 적용하였다.

Munkres 알고리즘을 적용한 무기할당 알고리즘의 성능을 확인하기 위해서 탐색 트리를 적용한 다른 두 알고리즘과 비교하였다. OWA 알고리즘은 최적의 할당 결과를 찾아주지만 시간적 측면에서 비효율적이며, GWA 알고리즘은 성능적인 면에서 MWA 알고리즘보다 떨어지고, 제약조건을 추가하면 성능 차이가 더 벌어지는 것을 확인하였다. 반면 제안한 MWA 알고리즘은 제약조건을 추가하는 것에 관계없이 polynomial time 내에 수행이 가능하고, OWA 알고리즘과 동일한 성능을 보이는 알고리즘으로 성능과 시간적 측면에서 모두 효율적이라는 것을 확인하였다.

본 논문에서 제안한 MWA 알고리즘은 무기할당 문제를 일반화하여 모델링 하였으나, 향후 실 무기체계에 적용하기 위해서는 좀 더 실질적이고 많은 제약조건들을 포함할 수 있는 모델링 기법이 제시되어야 할 것이다.

Reference

[1] Lloyd S. P., H. S. W., “Weapons Allocation is NP-complete”, Proceedings of the IEEE Summer Simulation Conference, Reno, Nevada, 1986.

[2] Cai Huaiping, Liu Jingxu, Chen Yingwu, Wang Hao, “Survey of the Research on Dynamic Weapon-Target Assignment Problem”, Journal of Systems Engineering and Electronics, Vol. 17, No. 3, pp. 559～565, 2006.

[3] Huang P. P. K., “An Autonomous Optimal Weapon Assignment Algorithm for Ship Self-Defence”, IEEE International Conference, Vol. 3, pp. 2545～3549, 1994.

[4] Khosla D., “Hybrid Genetic Approach for the Dynamic Weapon-Target Allocation Problem”, Proceedings of SPIE, 4396, pp. 248～263, 2001.

[5] Lee Zne-Jung, S-F S, Chou-Yuan Lee, “Efficiently Solving General Weapon-Target Assignment Problem by Genetic Algorithms with Greed Eugenics”, IEEE Journal on Systems and Cybernetics-Part B : Cybernetics, 2003.

[6] Li Hongrui, Miao Yan, “WTA with the Maximum Kill Probability Based on Simulated Annealing Algorithms”, The Special Committee of C2 and Computer of the Electronic Technology Academic Committee of China, Ship Engineering Society. Academia Conference, pp. 436～440, 2000.

[7] Rosenberger Jay M, Hee Su, Hwang A Y, et al., “The Generalized Weapon Target Assignment Problem”, The 10th International Command and Control Research and Technology Symposium on the Future of C2, McLean, VA, pp. 2～11, 2005.

[8] Harold W. Kuhn, “The Hungarian Method for the Assignment Problem”, Naval Research Logistics Quarterly, Kuhn's Original Publication, pp. 83～97, 1955.

[9] J. Munkres, “Algorithms for the Assignment and Transportation Problems”, Journal of the Society of Industrial and Applied Mathematics, Vol. 5, No. 1, pp. 32～38, 1957.