Sensor-Target and Weapon-Target Pairings Based on Auction Algorithm

Z. R. BOGDANOWICZ, N. P. COLEMAN

Armament Research, Development and Engineering Center (ARDEC)
Picatinny, NJ 07806
U.S.A.
http://www.pica.army.mil

Abstract: - Sensor-target and weapon-target pairings are important activities involved in planning and executing a course-of-action in a modern warfare. The outcome of today's combat operations may strongly depend on the intelligent usage of available sensors and weapons maximizing their effectiveness. The problem can be considered as an assignment optimization problem in mathematics. This problem is difficult because in the real world it involves many different factors and criteria to consider. We show that for practical sensor-target and weapon-target pairings a well-known auction algorithm should be considered the preferred choice.

Key-Words: - Assignment problem, auction algorithm, sensor-target pairing, weapon-target pairing.

1 Introduction

Sensor-target and weapon-target (or briefly sensor/weapon-target) pairings are challenging and difficult optimization problems. However, with faster computers and better algorithms it becomes more realistic and practical nowadays. Furthermore, the outcome of today's modern battles may strongly depend on the intelligent usage of available sensors and weapons maximizing their effectiveness. In general, these pairings consider many input types and the strategies of optimization might vary considerably. Hence, designing a single optimization algorithm for generic input is a hard problem. Efficiency of assigning weapons to targets might depend on the assignment of sensors to these targets, which further complicates this task. Sensor-target and weapon-target pairings, however, can be reduced to an assignment optimization problem, which is well known and studied in mathematics [3], [7-8], [10].

Let b_{jk} be a benefit of assigning weapon j to target k when sensor i is already assigned to target k, *i.e.,* $s_i \rightarrow t_k$. If there exists some other sensor-target pairing $s_{i'} \rightarrow t_k$, which implies benefit b'_{jk} of assigning weapon j to target k with $b'_{jk} \neq b_{jk}$, then such sensor/weapon-target pairing is called *dependent*. Otherwise sensor/weapon-target pairing is called *independent*. In this paper we consider both types of problems.

In today's battlefields many types of weapons often rely on supporting sensors. As an example of dependent sensor/weapon-target pairings, lasing of a target by a forward observer can guide a precision weapon to its precise destination. For independent sensor/weapon-target pairing, the weapons might still rely on sensors. For example, there might be just one type of sensor under consideration, and the weapons that rely solely on it. In such a case, any complete sensor-target pairing could result in identical weapon-target benefit matrix.

Let's now consider a two-step approach to the sensor/weapon-target pairing problem. In the first step, a preprocessing algorithm converts all the input information into two benefit matrices A,B, where each a_{ij} in A and each b_{ij} in B represents a benefit of assigning row i to column j. In the second step, an optimization algorithm assigns rows to columns in matrices A,B in such a way that the total benefit is maximized. In this paper we focus on the second step of the above approach. There are a number of optimal algorithms that can solve it and they are well documented – algorithms based on maximum matching in graphs [8], variants of the interior point algorithm [1,9,11], and the auction algorithm [2-7] to name a few. We show that for sensor/weapon-target pairing the *auction algorithm* should be considered the preferred choice. In addition, we back this up with performance results based on a simple *forward auction algorithm* implementation in Section 8.

2 Why Auction Algorithm

To solve an assignment optimization problem focused on sensors, weapons and targets, we might first consider if it makes sense to use an exact optimization algorithm vs. an approximate heuristic. Since the number of sensors, weapons and targets in

realistic battlefield scenarios should run up to hundreds, we estimate that an exact optimization algorithm should perform time-efficiently (i.e., in order of tens of seconds). Hence our attention should be focused on finding an exact optimization algorithm for the assignment problem for this range.

One of the better-documented exact optimization algorithms for an assignment problem is the auction algorithm [2-7]. For the above input size a well-implemented auction algorithm should run in order of seconds, as it is verified in section 7. Furthermore, the bidding and assignment phases of the auction algorithm are highly parallelizable [4], which makes it scalable. That is, the bidding and the assignment can be carried out for all sensors, weapons and targets simultaneously, which could extend the range of input to thousands of sensors, weapons and targets and beyond.

Finally, the nature of sensor-target and weapon-target pairings should allow a benefit scaling, which could produce matrices A,B with all integral benefits $a_{ij,}$, b_{ij} (Section 7). This in turn could further improve the performance of the auction-based algorithms. In fact, this is the key for an efficient implementation of any auction algorithm.

For much larger input sizes one could also consider variants of the interior point algorithm [1, 11]. However, such inputs would be extremely rare for sensor/weapon-target pairing in the real world. In addition, one could still address this class of problems with the parallel implementation of an auction algorithm.

3 Sensor/Weapon-Target Pairing Problem for Auction Algorithm

Let a_{ij} be a value of assigning sensor i into target j, $x_{ij}=1$ indicate that sensor i is assigned into target j, and otherwise $x_{ij}=0$. Let b_{ij} be a value of assigning weapon i into target j, $y_{ij}=1$ indicate that weapon i is assigned into target j, and otherwise $y_{ij}=0$. We have the following sensor/weapon-target pairing mathematical formulation.

$$max \sum_{i,j} (a_{ij} \cdot x_{ij} + b_{ij} \cdot y_{ij}) \quad (1)$$

subject to

$$\sum_{i} x_{ij} \le 1 \quad (2)$$

$$\sum_{j} x_{ij} \le 1 \quad (3)$$

$$\sum_{i} y_{ij} \le 1 \quad (4)$$

$$\sum_{j} y_{ij} \le 1 \quad (5)$$

The input to auction algorithm are matrices $A[a_{ij}]_{n_1 n_3}$, $B=[b_{ij}]_{n_2 n_3}$, where a_{ij} represents a benefit of assigning row i to column j in A (i.e., benefit of assigning corresponding sensor i to target j), and b_{ij} represents a benefit of assigning row i to column j in B (i.e., benefit of assigning corresponding weapon i to target j).

In general, the number of sensors n_1 or weapons n_2 does not equal to the number of targets n_3. If $n_1 \neq n_3$ and/or $n_2 \neq n_3$ then the input A, B to an auction algorithm can be easily translated into A', B' with $n_1=n_2=n_3=n$ by appending either rows with 0 entries (corresponding to phantom sensors/weapons) or columns with 0 entries (corresponding to phantom targets). Hence without loss of generality consider A, B as symmetric matrices of rank $n \times n$ for analysis of auction algorithm performance.

There are two cases to consider for auction algorithm performance based on whether sensor/weapon-target pairings are dependent or not.

4 Independent Sensor/Weapon-Target Pairings

If sensor/weapon-target pairings are independent then we can translate (1) to the following optimization:

$$max \sum_{i,j} a_{ij} \cdot x_{ij} + max \sum_{i j} b_{ij} \cdot y_{ij} \quad (6)$$

with constraints (2) through (5) staying the same.

An auction-based algorithm for independent sensor/weapon-target pairings can be presented as follows:

Step 1: Generate benefit matrices A, B based on every possible sensor-target and weapon-target pair.

Step 2a: Execute an auction algorithm for sensor-target pairings based on A.

Step 2b: Execute an auction algorithm for weapon-target pairings based on B.

Let ε be a minimum increase of bid cost for a sensor/target if an auction algorithm assigns a sensor/weapon to that target in its iteration. Assuming integral benefit matrices A,B (i.e., all a_{ij}, b_{ij} being integers), the auction algorithm guarantees that the feasible result of optimization is optimal if $\varepsilon < 1/n$ [2].

Consider now the worst-case running time complexity of accomplishing an optimal assignment.

The assignment problem can be modeled with a bipartite graph $G=(V,E)$, where the number of vertices $|V(G)|=2n$ and the number of edges $|E(G)|=m$ in G. Let $C=max_{(i,j)\in E(G)}c_{ij}$, where c_{ij} represents a benefit of assigning sensor/weapon i to target j. The total number of iterations in which a target receives a bid is no more than C/ε. In addition, an auction algorithm can be implemented in such a way that its iteration involves a bid by a single sensor/weapon. So, the total number of iterations is no more than $2n$ times C/ε, and since every bid requires $O(n)$ operations, the worst running time of the algorithm is

$$O(n^2C/\varepsilon) \quad (7)$$

If all benefits $a_{ij}\in A$, $b_{ij}\in B$ are integers (which we can accomplish in most cases by scaling up every $a_{ij}\in A$ and every $b_{ij}\in B$ by appropriately large integer), and $\varepsilon<1/n$, then according to [4] the worst time complexity is

$$O(nm \log(nC)). \quad (8)$$

5 Dependent Sensor/Weapon-Target Pairings

If sensor/weapon-target pairings are dependent then the benefit matrices A,B (for assigning sensors and weapons to targets respectively) can be translated into benefit matrix H. In H each row corresponds to a unique sensor/weapon combination and each column corresponds to a target. That is, h_{ij} represents a benefit of assigning i'th distinct sensor/weapon combination to target j. Such a translation requires $O(n^3)$ operations. So H has n^2 rows and n columns, and represents the only input to our modified auction algorithm, which executes predominantly as a standard auction algorithm. It assigns n out of n^2 rows to n columns in H with the following exceptions.

a) If sensor s_i and weapon w_j are currently assigned to target t_k, based on the best bid (i.e., $s_iw_j\rightarrow t_k$), then $s_{i'}w_{j'}\rightarrow t_{k'}$ is not considered for assignment to target k', $k' \neq k$ if either $i=i'$ or $j=j'$. This assures that a sensor/weapon or target is not assigned more than once in an optimal solution.

b) Based on the best $s_iw_j\rightarrow t_k$ assignment a second best assignment for target k', $k' \neq k$ is determined by $s_{i'}w_{j'}\rightarrow t_{k'}$, where either $i'=i$ or $j'=j$ depending on implementation. This allows calculation of a penalty cost for auction bids.

So, a modified auction-based algorithm for dependent sensor/weapon-target pairings can be presented as follows:

Step 1: Generate benefit matrix H based on every possible triplet combination $< s_i, w_j, t_k >$.

Step 2: Execute a modified auction-based algorithm for sensor/weapon-target pairings based on matrix H and rules (a) and (b).

As was the case with independent sensor/weapon-target pairings, the total number of iterations in which a target receives a bid is no more than C/ε. In addition, our modified auction algorithm can be implemented in such a way that its iteration involves a bid by a pair of a single weapon along with the best available (unassigned) sensor. So, the total number of iterations is no more than n times C/ε, and since every bid requires now $O(n^2)$ operations, the worst running time of the algorithm is

$$O(n^3C/\varepsilon) \quad (9)$$

6 Auction Algorithm Anomaly

The running time of the auction algorithm is not always monotonic in respect to the ranks of A, B. In the best scenario an auction algorithm requires just $O(m)$ comparisons/assignments to arrive at the optimal solution.

The following example illustrates an apparent anomaly, where the number of iterations and the number of benefit comparisons actually decreases as the new sensors and targets are introduced to a current scenario in the battlefield. Let the initial benefit matrix A be as illustrated in Figure 1.

$$A=\begin{pmatrix} 9 & 6 & 2 \\ 8 & 4 & 1 \\ 7 & 2 & 0 \end{pmatrix}$$

Fig. 1 – Initial benefit matrix

Let sensors s_1, s_2, s_3 correspond to rows *1, 2, 3*, and targets t_1, t_2, t_3 correspond to columns *1,2,3* in A. If the search order in an auction algorithm starts with the first row and column, and continues through the consecutive rows/columns, then the corresponding sensor-target assignment sequence is as follows:

$$s_1 \rightarrow t_1,\ s_2 \rightarrow t_1,\ s_3 \rightarrow t_1,\ s_1 \rightarrow t_2,\ s_2 \rightarrow t_1,\ s_3 \rightarrow t_1,$$
$$s_2 \rightarrow t_2,\ s_1 \rightarrow t_2,\ s_2 \rightarrow t_3 \qquad (10)$$

Clearly, it requires 9 sensor-target assignments and consequently 9*3 =27 benefit comparisons. Consider now augmented A (i.e., A') with two additional rows corresponding to sensors s_4, s_5, and with two additional columns corresponding to targets t_4, t_5 (Figure 2).

For the same search order as before, the corresponding sensor-target assignment sequence is as follows:

$$s_1 \rightarrow t_4,\ s_2 \rightarrow t_5,\ s_3 \rightarrow t_1,\ s_4 \rightarrow t_2,\ s_5 \rightarrow t_3 \qquad (11)$$

In this case the optimization requires 5 sensor-target assignments, and consequently only 5*5 =25 benefit comparisons. This is due to less required re-assignments in (11). Thus, sequence (11) requires two less benefit comparisons than the initial sequence (10). Note, the sequences (10), (11) can be verified based on [2].

$$A' = \begin{pmatrix} \mathbf{9} & \mathbf{6} & \mathbf{2} & 10 & 1 \\ \mathbf{8} & \mathbf{4} & \mathbf{1} & 1 & 10 \\ \mathbf{7} & \mathbf{2} & \mathbf{0} & 0 & 0 \\ 3 & 4 & 0 & 0 & 0 \\ 3 & 4 & 5 & 2 & 1 \end{pmatrix}$$

Fig. 2 – Augmented benefit matrix

7 Efficiency of Scaling

The key to a good performance of an auction algorithm for sensor/weapon-target pairing problem is intelligent scaling of matrices A,B and H. An efficient way to do it for independent sensor/weapon-target pairing would be as follows. A preprocessing algorithm converts all the sensor/weapon-target related input information into two nxn benefit matrices A,B, where each a_{ij} in A and each b_{ij} in B represents a benefit of assigning sensor/weapon i to target j with at most k places after the decimal point. Considering the fact that many inputs have to be taken into account in the real world for a sensor/weapon-target pairing scenario, it makes sense not to consider too many places after the decimal point. Then matrices A,B are converted into matrices A', B' where $a'_{ij} = n \cdot a_{ij} \cdot 10^k$ and $b'_{ij} = n \cdot b_{ij} \cdot 10^k$. In addition, a minimum bidding increment parameter ε is set to 0.99 (i.e., in general $\varepsilon < 1$ must be satisfied). Consequently, this set-up assures that any complete assignment generated by an auction algorithm is optimal. So, the optimization can be terminated after the first complete assignment is obtained. A similar approach can be applied to dependent sensor/weapon-target pairings. In this case matrix H is converted into matrices H', where $h'_{ij} = n \cdot h_{ij} \cdot 10^k$. A minimum bidding increment parameter ε again is set to 0.99.

8 Computational Results and Conclusions

We implemented a simple forward auction with the benefit scaling algorithm in a Linux RedHat 9.0 environment on a PC with Intel® Pentium ® 4 CPU 1500 MHz. Our implementation supports the input of up to 1,000 sensors/weapons by 1,000 targets. We conducted performance experiments for independent sensor/weapon-target pairings with inputs of up to 200 sensors/weapons by 200 targets.

The optimal solution resulted in an improvement of up to 12% over the initial score obtained by our ad hoc heuristics. The average improvement resulted in 3.3%, and the average execution time of the auction algorithm was 2.8s. In addition, the average execution time per target was 0.03s. We also observed that the benefit of getting an optimal solution tends to increase as the total number of sensors, weapons and targets increases.

Based on these results we conclude that for dependent sensor/weapon-target pairing problems, the distributed implementation of the auction algorithm could be a critical factor in satisfying real-world optimization scenarios, and could yield even better performance. Such implementation is well suited for the auction algorithm as indicated in [4].

References

[1] I. Adler, N. Karmarkar, M. Resende, G. Veiga, An Implementation of Karmarkar Algorithm for Linear Programming, *Math. Programming* **44** (1989), pp. 297-335.

[2] D. P. Bertsekas, *Linear Network Optimization: Algorithms and Codes*, The MIT Press, 1992.

[3] D. P. Bertsekas, The Auction Algorithm for Assignment and Other Network Flow Problems, *Interfaces* **20** (1990), pp. 133-149.

[4] D. P. Bertsekas, Auction Algorithms for Network Flow Problems: A Tutorial Introduction, *Comp. Optimiz. and Appl.* **1** (1992), pp. 7-66.

[5] Z. Bogdanowicz and N. Coleman, Efficient methodology and robust infrastructure for assigning weapons to targets, *AMCS Proceedings*, CSREA Press, (2004), pp. 289-295.

[6] Z. Bogdanowicz, N. Coleman, and S. Kaniyantethu, Combat Decision Support Subsystem for Optimal Weapon-Target Assignment, *DCDIS Proceedings*, Watam Press, (2005), pp. 11-15.

[7] D. A. Castanon, Reverse Auction Algorithms for Assignment Problems, *Algorithms for Network Flows and Matching – American Math. Soc.* (1993), 407-429.

[8] Z. Galil, Efficient Algorithms for Finding Maximum Matchings in Graphs. *ACM Computing Surveys*, **18** (1986), 23-38.

[9] N. Karmarkar, A New Polynomial-time Algorithm for Linear Programming, *Combinatorica* **4** (1984), 373-395.

[10] L. Lovász and M. Plummer. *Matching Theory*. North-Holland, Amsterdam, 1986.

[11] M. J. Todd, A Low Complexity Interior Point Algorithm for Linear Programming, *SIAM Journal of Optimization* **2** (1992), 198-209.