Weapon Target Assignment problem

Guanda Li

Institute of Computational Mathematics and Scientific/Engineering Computing,
Academy of Mathematics and Systems Science,
Chinese Academy of Sciences

December 20, 2022

Contents

Introduction

Background

- Weapon Target Assignment problem is a problem in the military field
- The basic problem is to consider using m weapons to attack n targets, in order to minimize the weighted survival probability of all targets.
- The problem is proved to be NP-complete

Basic formulation

- $I = \{1, \cdots, m\}$, weapon set.
- $J = \{1, \cdots, n\}$, target set.
- $p_{ij} \in [0, 1]$, probability that i hits j
- V_j, weight of the target j.
- x_{ij}, decision variables, whether weapon i attack j.

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} V_j \left(1 - \prod_{i=1}^{m} (1 - p_{ij})^{x_{ij}} \right) \qquad \text{(S0)} \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \leq 1 \quad \forall\, i \in I, \\ & x_{ij} \in \{0, 1\} \quad \forall\, j \in J,\ i \in I. \end{aligned}$$

Basic formulation

- $I = \{1, \cdots, m\}$, weapon set.
- $J = \{1, \cdots, n\}$, target set.
- $p_{ij} \in [0,1]$, probability that i hits j
- V_j, weight of the target j.
- x_{ij}, decision variables, whether weapon i attack j.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} V_j \left(\prod_{i=1}^{m} (1-p_{ij})^{x_{ij}} \right) & \text{(S0')} \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \le 1 \quad \forall\, i \in I, \\ & x_{ij} \in \{0,1\} \quad \forall\, j \in J,\ i \in I. \end{aligned}$$

Weapon target assignment problem

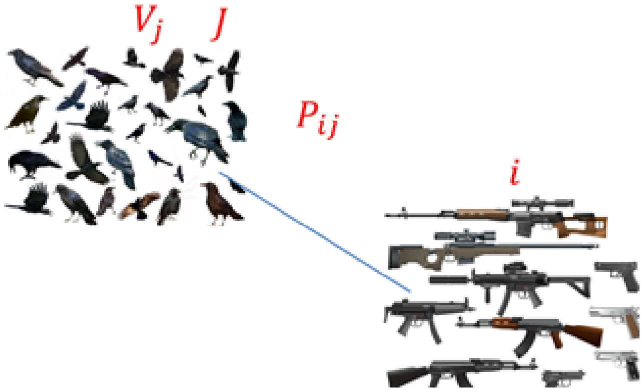

Figure: WTA problem

History

- Manne (1958)
  Give the basic formulation of the problem. Solve the problem with linear approximations to problems, exactly solve small-scale problems.
- denBroeder (1959) , Hosein(1989)
  solve a simpler model assume that all the weapon has the same probability of hitting the same target.
- Lloyd Witsenhause (1986)
  Prove the problem is NP-complete.
- Johansson Falkman (2009)
  Use search algorithm solve the problem with 9 weapons and 8 targets in 13 minutes.
- Rosenberger et al. (2005), Ahuja et al. (2007), Kline (2017)
  Use branch-and-bound frame work to solve the WTA problem. It takes 16.2 hours to solve the model with size 80 weapons and 80 targets.

History

- Lu(2021)
  Transform the problem into a huge size linear programming and use column enumeration to solve the problem.
- Anderson(2022)
  Use lower linear approximation of the objective function in a branch-and-bound framework.
- These two new techniques significantly improved the computation efficiency. Both of them could solve the problem with size 400 weapons and 400 targets in a few minutes.

Formulations and Algorithms

WTA model 1

- Compared to the basic model, allows w_i weapons for each weapon type.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} V_j \left(\prod_{i=1}^{m} (1 - p_{ij})^{x_{ij}} \right) \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \le w_i \quad \forall\, i \in I, \\ & x_{ij} \in \mathbb{Z}^+ \quad \forall\, j \in J,\ i \in I. \end{aligned} \tag{S1}$$

WTA model 2

- For convenience, assume that any weapon has the same probability of hitting the same target.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} V_j (1-p_j)^{\sum_{i=1}^{m} x_{ij}} \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \le w_i \quad \forall\, i \in I, \\ & x_{ij} \in \mathbb{Z}^+ \quad \forall\, j \in J,\ i \in I. \end{aligned} \tag{S2}$$

WTA model 3

- Logarithm of the objective function in S1

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{n} V_j \exp\left(\sum_{i=1}^{m} x_{ij} \ln(1-p_{ij})\right) \\
\text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \le w_i \quad \forall\, i \in I, \\
& x_{ij} \in \mathbb{Z}^{+} \quad \forall\, j \in J,\ i \in I.
\end{aligned}
\tag{S3.1}
$$

WTA model 3

- Replaced the objective function with y_j to get the following model.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} V_j e^{y_j} \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \le w_i \quad \forall\, i \in I, \\ & \sum_{i=1}^{m} \ln(1-p_{ij}) x_{ij} = y_j \quad \forall\, j \in J \\ & x_{ij} \in \mathbb{Z}^+ \quad \forall\, j \in J,\ i \in I. \end{aligned} \tag{S3.2}$$

- The model is equivalent to model S1, but it is more convenient for the solver to calculate.
- Kline et al.(2017b) points out that use the commercial solver BARON to solve the problem, this form can increase the correct rate by 21%

WTA model 4

- Limit the x_{ij} to bianry variable.
- For $(1 - p_{ij}x_{ij}) = (1 - p_{ij})^{x_{ij}},\ x \in \{0, 1\},$ The problem can be transformed into the following form.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} V_j \left(\prod_{i=1}^{m} (1 - p_{ij}x_{ij}) \right) \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \leq 1 \quad \forall\ i \in I, \\ & x_{ij} \in \{0, 1\} \quad \forall\ j \in J,\ i \in I. \end{aligned} \tag{S4}$$

- This conversion will cause the relaxation of the problem from a convex from to a non-convex form.

WTA model 5

- Transform the problem into a knapsack problem

$$\begin{aligned} \min \quad & \sum_{j=1}^{n}\sum_{x=1}^{m} c_{ij}x_{ij} \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \le 1 \quad \forall\, i \in I, \\ & \sum_{i=1}^{m} x_{ij} \le 1 \quad \forall\, j \in J \\ & x_{ij} \in \{0,1\} \quad \forall\, j \in J,\ i \in I. \end{aligned} \tag{S5}$$

- This method removes the coupling when multiple weapons strike the same weapon, transforms the difficult objective function into a linear function.

Outer Approximation

Basic formulation

- $I = \{1, \cdots, m\}$, weapon set.
- $J = \{1, \cdots, n\}$, target set.
- $p_{ij} \in [0, 1]$, probability that i hits j
- V_j, weight of the target j.
- x_{ij}, decision variables, whether weapon i attack j.

$$\min \quad \sum_{j=1}^{n} V_j \left(\prod_{i=1}^{m} (1 - p_{ij})^{x_{ij}} \right) \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} x_{ij} \leq 1 \quad \forall\, i \in I, \tag{2}$$

$$x_{ij} \in \{0, 1\} \quad \forall\, j \in J,\ i \in I. \tag{3}$$

Convexity of the objective function

$$f_j(x) = \prod_{i=1}^{m} (1 - p_{ij})^{x_{ij}} \quad \forall\, j \in J$$

Hessian matrix H:

$$\frac{\partial f_j(x)}{\partial x_{aj} \partial x_{bj}} = \ln(1 - p_{aj}) \ln(1 - p_{bj}) f_j(x)$$

$$H = f(x) \begin{bmatrix} \ln(1-p_{1j})\ln(1-p_{1j}) & \ln(1-p_{1j})\ln(1-p_{2j}) & \cdots & \ln(1-p_{1j})\ln(1-p_{mj}) \\ \ln(1-p_{2j})\ln(1-p_{1j}) & \ln(1-p_{2j})\ln(1-p_{2j}) & \cdots & \ln(1-p_{2j})\ln(1-p_{mj}) \\ \vdots & \vdots & \ddots & \vdots \\ \ln(1-p_{mj})\ln(1-p_{1j}) & \ln(1-p_{mj})\ln(1-p_{2j}) & \cdots & \ln(1-p_{mj})\ln(1-p_{mj}) \end{bmatrix}$$

Convexity of the objective function

Let

$$l = \begin{bmatrix} \ln(1-p_{1j}) & \ln(1-p_{2j}) & \cdots & \ln(1-p_{mj}) \end{bmatrix}$$

Hessian matrix is :

$$H = f(x) l \cdot l^T$$

The Hessiann matrix is a rank-one matrix so the objective function is convex.

Transformed model

Objective function

$$\sum_{j=1}^{n} a_j \left(\prod_{i=1}^{m} (1 - p_{ij})^{x_{ij}} \right)$$

By introducing auxiliary variables, the nonlinear term can be transformed from the objective function into the constraint.
If we take η_j as the auxiliary variables to $\prod_{i=1}^{m}(1 - p_{ij})^{x_{ij}}$ the model can be transformed to

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} a_j \eta_j && \text{(S0')} \\ \text{s.t.} \quad & \eta_j \geq \prod_{i=1}^{m} (1 - p_{ij})^{x_{ij}}, \quad \forall\, j \in J \\ & \sum_{j=1}^{n} x_{ij} \leq 1 \quad \forall\, i \in I, \\ & x_{ij} \in \{0, 1\} \quad \forall\, j \in J \quad i \in I. \end{aligned}$$

Basic idea of outer approximation

If $f(x)$ is a convex function, then for any point x^* in the feasible region, we have

$$f(x) \geq f(x^*) + \nabla f(x^*)(x - x^*)$$

Therefore, if the constraint of the original problem is $\eta \geq f(x)$, then

$$\eta \geq f(x) \geq f(x^*) + \nabla f(x^*)(x - x^*)$$

must be correct

- For any given point, a linear constraint can be introduced to ensure that the feasible region satisfies the constraint.
- The outer approximation method try to replace a nonlinear constraint by some linear constraints, but guarantees the models are equivalent.

Outer approximation constraint

take $f_j(x) = \prod_{i=1}^{m}(1-p_{ij})^{x_{ij}} \quad j \in J$, then for any given $\bar{x}$ in feasible domain, we have

$$\begin{aligned}\nabla f(\bar{x})(x-\bar{x}) &= f(\bar{x})\sum_{i=1}^{m}\ln(1-p_{ij})(x_{ij}-\bar{x_{ij}}) \\ &= f(\bar{x})\sum_{i=1}^{m}\ln(1-p_{ij})x_{ij} - f(\bar{x})\sum_{i=1}^{m}\ln(1-p_{ij})\bar{x_{ij}}\end{aligned}$$

- We denote this constraint as the outer approximation constraint.

Outer approximation model

Let X donates the set of all integer feasible solutions in the problem, the model can be described as.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} a_j \eta_j && \text{(OA)} \\ \text{s.t.} \quad & \eta_j \geq f(\bar{x}) \sum_{i=1}^{m} \ln(1-p_{ij})(x_{ij}-\bar{x}_{ij}) + f(\bar{x}), \quad \forall\, j \in J,\ \bar{x} \in X \\ & \sum_{i=1}^{m} x_{ij} \leq 1 \quad \forall\, j \in J, \\ & x_{ij} \in \{0,1\} \quad \forall\, j \in J \quad i \in I. \end{aligned}$$

the number of outer approximation constraints is very large. The restricted model only take care of some of them.

Idea of outer approximation method

We assume that $\hat{X} \subseteq X$.

$$\begin{aligned}
\min \quad & \sum_{j=1}^{n} a_j \eta_j \\
\text{s.t.} \quad & \eta_j \geq f(\bar{x}) \sum_{i=1}^{m} \ln(1 - p_{ij})(x_{ij} - \bar{x_{ij}}) + f(\bar{x}), \quad j \in J, \ \bar{x} \in \hat{X} \\
& \sum_{i=1}^{m} x_{ij} \leq 1 \quad j \in J, \quad x_{ij} \in \{0,1\} \quad j \in J \quad i \in I
\end{aligned}$$

Algorithm of outer approximation

1. Remove all outer approximation constraints and give the current optimal solution x, η
2. Check whether the current optimal solution can satisfy all the outer approximation constraints. if true, the iteration terminates and end the execution.
3. Otherwise, choose to outer approximation constraint that has been violate most and add it to the restricted model, resolve the model and go to step 2.

Choose violated constraint

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{n} a_j \eta_j \\
\text{s.t.} \quad & \eta_j \geq f(\bar{x}) \sum_{i=1}^{m} \ln(1 - p_{ij})(x_{ij} - \bar{x}_{ij}) + f(\bar{x}), \quad j \in J,\ \bar{x} \in X \\
& \sum_{i=1}^{m} x_{ij} \leq 1 \quad j \in J, \quad x_{ij} \in \{0,1\} \quad j \in J \quad i \in I
\end{aligned}
$$

- Unless an optimal solution has been found, the solution to the restricted problem must bring a violation of the outer approximation constraint.
- When obtaining a solution that x is in the feasible region, if $\eta < f_j(x)$, this causes infeasible. the infeasible point can be excluded by adding an outer approximation constraint.

Numerical results

Weapon-Target Assignment Problem Instances [1]

- SET1: Only to separate integer infeasible solutions.
- SET2: Also to separate fractional infeasible solutions.

	SET1		SET2	
$\lvert I\rvert \times \lvert J\rvert$	time(s)	nodes	time(s)	nodes
5×5	0.20	2	0.20	2
10×10	0.01	9	0.01	3
20×20	0.38	1680	0.28	69
30×30	152.49	491403	40.40	16932
40×40	3600.00+	–	77.14	38967
50×50	3600.00+	–	3600.00+	–

[1] Emrullah SONUC, Baha SEN, and Safak BAYIR, A Parallel Simulated Annealing Algorithm for Weapon-Target Assignment Problem, International Journal of Advanced Computer Science and Applications, 8(4), 2017

Columnn Generation

Basic Idea

- Some linear programming problems have too many columns (variables), making it difficult to solve
- Use only some of the variables at the beginning of the algorithm and assume all the other variables are 0
- Variables that have the potential to improve the objective function are iteratively added to the model.
- Once it can be proven that adding new variables will no longer improve the value of the objective function, the iterative process is terminated and an optimal solution is obtained.

How to use column generation in WTA Problem

- **Basic Idea**: By listing all the weapon assignment scenarios S, the problem is transformed into a linear programming problem.
- Assuming that there are m weapons, for any target j, each weapon can choose to attack j or not attack j, so there are 2^m different attack schemes.
- **an example**: There are a total of 8 weapons, and the attack plan using No. 1, 3, and 6 weapons is recorded as $s_{[1,0,1,0,0,1,0,0]}$, $|S| = 2^8 = 256$
- n_{si}: bianry variable, indicates whether to enable weapon i in the sth scene. In the above example, $n_{s1} = 1, \ n_{s2} = 0$
- $q_{js} = a_j \prod_{i=1}^{m}(1 - n_{si} \cdot p_{ij})$: weighted probability of the plan s to hit the target j, For example, q_{3s} is to use the No. 1, 3, and 6 weapons to hit the target 3 and multiply the probability of destroying the target j by the weight of the target j.

Transformed formulation

$$\min \quad \sum_{j=1}^{n}\sum_{s=0}^{2^m-1} q_{js}y_{js} \tag{CG}$$

$$\text{s.t.} \quad \sum_{j=1}^{n}\sum_{s=0}^{2^m-1} n_{si}y_{js} \le 1 \qquad \forall\, i \in I$$

$$\sum_{s=0}^{2^m-1} y_{js} = 1 \qquad \forall\, j \in J$$

$$y_{js} \in \{0,1\} \qquad \forall\, j \in J,\ s \in S$$

- $I = \{1, \cdots, m\}$, weapon set
- $J = \{1, \cdots, n\}$, target set
- $S = \{1, \cdots, 2^m\}$, scene set

- n_{si}: weather scene s use weapon i
- y_{js}: Whether use s to arrack target j
- q_{js}: weighted destruction probability of using s to j

Transformed formulation continuous

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{n} \sum_{s=0}^{2^m-1} q_{js} y_{js} && \text{(CG)} \\
\text{s.t.} \quad & \sum_{j=1}^{n} \sum_{s=0}^{2^m-1} n_{si} y_{js} \le 1 && \forall\ i \in I \\
& \sum_{s=0}^{2^m-1} y_{js} = 1 && \forall\ j \in J \\
& y_{js} \in \{0,1\} && \forall\ j \in J,\ \ s \in S
\end{aligned}
$$

- **objective function**: minimize the weighted destruction probabilities.
- **first constraint**: each weapon can only attack one target.
- **second constraint**: assign exactly one scene for each target

Column enumeration

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{n} \sum_{s=0}^{2^m-1} q_{js} y_{js} && \text{(CG)} \\
\text{s.t.} \quad & \sum_{j=1}^{n} \sum_{s=0}^{2^m-1} n_{si} y_{js} \le 1 && \forall\, i \in I \\
& \sum_{s=0}^{2^m-1} y_{js} = 1 && \forall\, j \in J \\
& y_{js} \in \{0,1\} && \forall\, j \in J,\ s \in S
\end{aligned}
$$

- This idea is from Lu(2021)
- The basic idea of the column enumeration method is to enumerate all columns in a smarter way
- Two techniques are used in the article: weapon number bounding and weapon domination
- weapon number bounding: Scenarios with too few or too many weapons are give no improvement to the objective function.

LP Relaxition

$$\begin{aligned}
\min \quad & \sum_{j=1}^{n} \sum_{s=0}^{2^m-1} q_{js} y_{js} && \text{(CG-LP)} \\
\text{s.t.} \quad & \sum_{j=1}^{n} \sum_{s=0}^{2^m-1} n_{si} y_{js} \leq 1 && \forall\ i \in I \\
& \sum_{s=0}^{2^m-1} y_{js} = 1 && \forall\ j \in J \\
& y_{js} \geq 0 && \forall\ j \in J,\ \ s \in S
\end{aligned}$$

- LP Relaxition, the third constraint can be changed into $y_{js} \geq 0$
- Linear program contains $n \times 2^m$ columns
- Only some of the columns (variables) are taken, because in the optimal solution of linear programming, at most $m + n$ variables are not 0, that is, the dual constraints corresponding to other variables are not active.

Dual Problem

$$\begin{aligned} \max \quad & \sum_{i=1}^{m} u_i + \sum_{j=1}^{n} v_j \\ \text{s.t.} \quad & \sum_{i=1}^{m} x_{is} u_i + v_j \le q_{js} \quad \forall\, (s,j) \in X \\ & u_i \le 0, \quad v_j \text{ free} \end{aligned} \tag{CG-Dual}$$

- If $X = \{(s,j) | s \in S, j \in J\}$, each element in X corresponds to a constraint.
-
- Selecting some columns in the original problem is equivalent to selecting some rows in the dual problem.

Restricted Dual Problem

$$\begin{aligned} \max \quad & \sum_{i=1}^{m} u_i + \sum_{j=1}^{n} v_j && \text{(CG-Dual-R)} \\ \text{s.t.} \quad & \sum_{i=1}^{m} x_{is} u_i + v_j \leq q_{js} \quad \forall (s,j) \in \hat{X} \\ & u_i \leq 0, \quad v_j \text{ free} \end{aligned}$$

- Only select some constraints, that is, only consider the constraints generated by the some (s, j) in $\hat{X} \subseteq X$
- Solve to get u^*, v^* and bring into the original dual, if all the constraints are satisfied, it must be the optimal solution.
- Otherwise, choose the constraint that violates the most, that is, the largest $\sum_{i=1}^{m} x_{is} u_i + v_j > q_{js}$.

Subproblem

$$\begin{aligned} \min \quad & a_j \prod_{i=1}^{m} (1 - p_{ij} x_{is}) - \sum_{i=1}^{m} x_{is} u_i - v_j \\ \text{s.t.} \quad & j \in J, \quad s \in S \end{aligned} \tag{CG-sub}$$

- Choose the constraint that violates the most, that is, the largest $\sum_{i=1}^{m} x_{is} u_i + v_j > q_{js}$.
- Using the definition of q_{js} to get the above sub-problems.
- The problem can be separated and then solved for each given target j.
- It is intuitive that using each weapon to attack requires a cost, we try to balance the cost of the weapon and the probability of destroying the target.

Motivation to use column generation

- Huge improvement in computational result for column enumerations
- The article on column enumeration mentions that column generation subproblems is hard to solve due to non-linearity
- The outer approximation method can be used in subproblems

Future Work

Use column generation in $S1$

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} V_j \left(\prod_{i=1}^{m} (1 - p_{ij})^{x_{ij}} \right) \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \le w_i \quad \forall\, i \in I, \\ & x_{ij} \in \mathbb{Z}^+ \quad \forall\, j \in J,\ i \in I. \end{aligned} \tag{S1}$$

More comlicated model

- Dynamic WTA problem.
- muti-objective WTA problem.
- Sensor WTA problem.

Thank you!