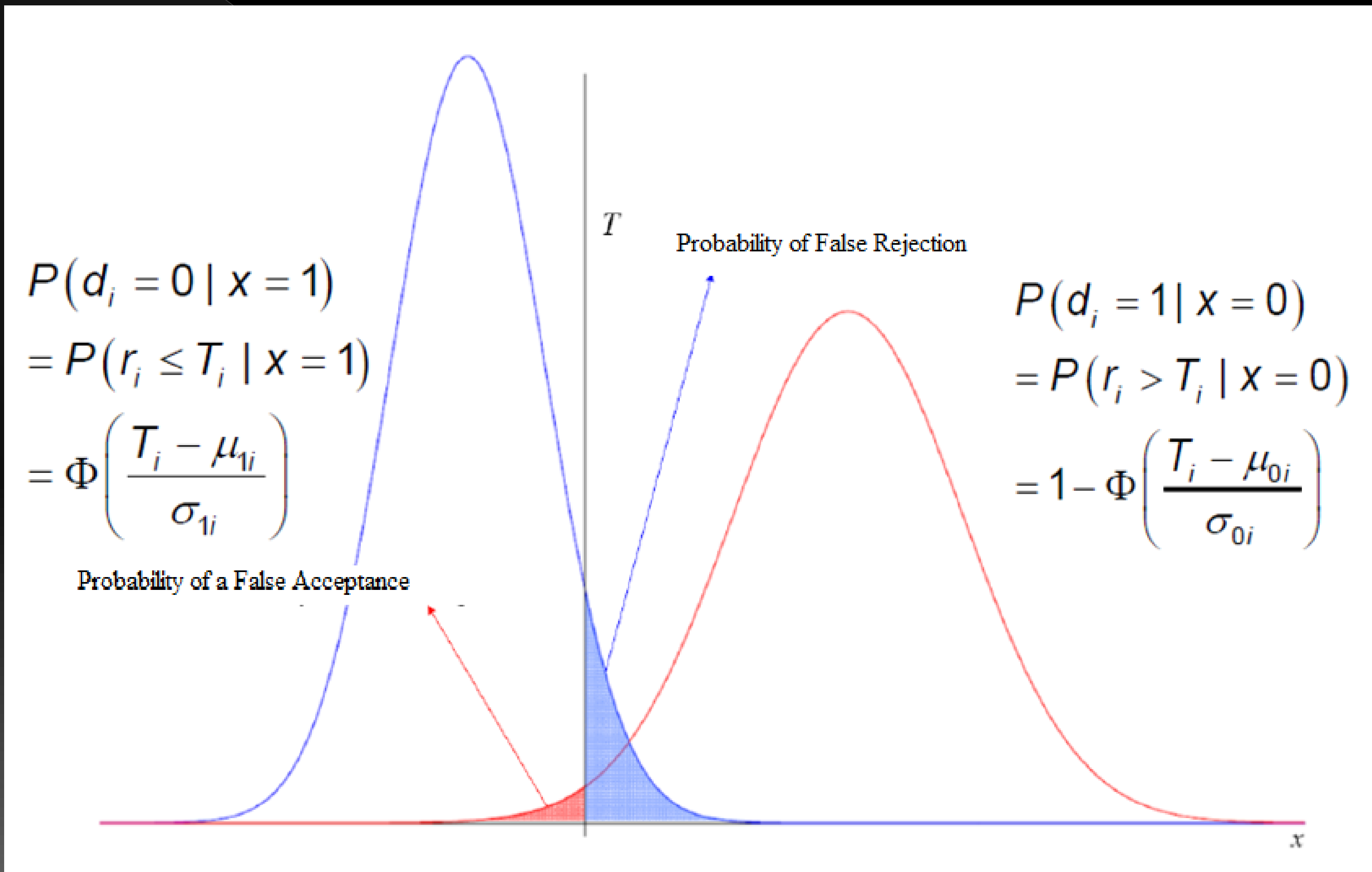
Optimum Strategies of Container Inspection at Port-of-Entry

Somair Malik
Rutgers University
Mentors: Dr. E.A Elsayed, Dr. Minge Xie

Port-of-Entry Problem Basics

- Inspection system consists of *k* stations
- Station *j* corresponds to risk category *j* and is effective in detecting undesired material of type *j*.
- There are *n* containers and each one is associated with a risk profile.
- A risk profile is the perceived likelihood of a container including unacceptable material of each of the *k* defined categories
- If container passes inspection stations: acceptable cargo
- If container does not pass inspection station: illegal cargo

Possible Errors

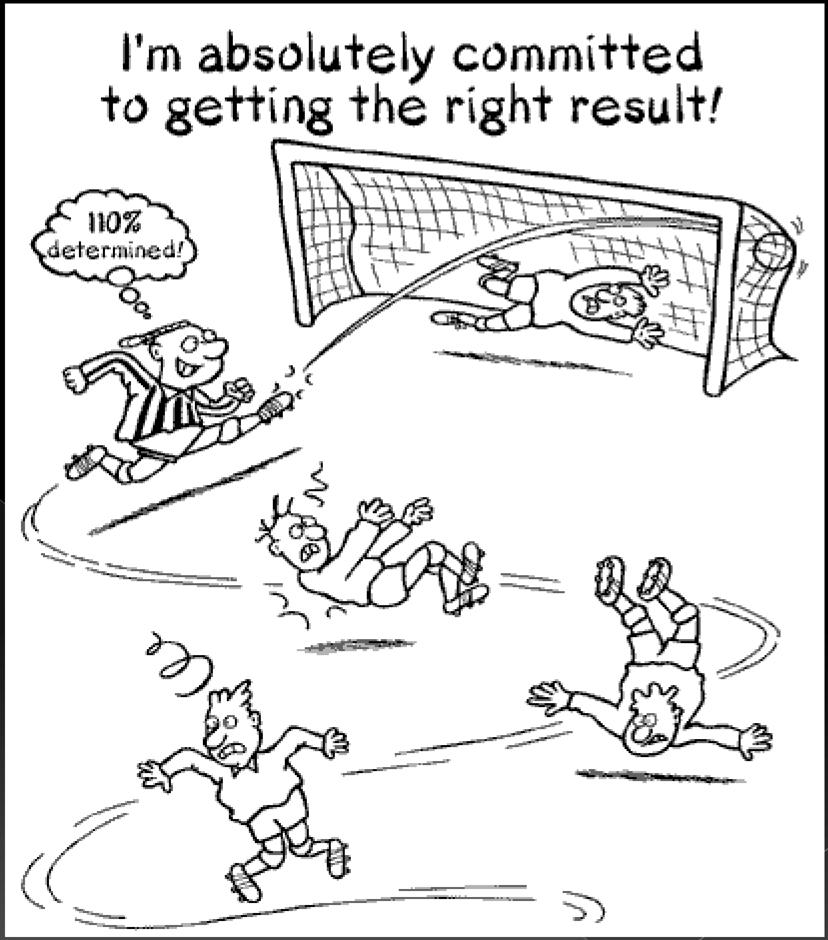


Important Factor

- Due Times: deadline for a container to be released, either for pickup by the receiving company or transport to a vessel for travel to the next destination
- Delays = financial loss for receiving companies and overtime costs for the inspection provider.
- Delays determine tardiness values.

Goal

Determine an optimal allocation by selecting the inspection operations that have the greatest effect on **false acceptance probabilities**, while weighing the effect on the optimal schedule and **tardiness**.



$$\min \quad 1 - w_T \sum_{i=1}^{n} \sum_{j=1}^{k} \pi_{ij} \left[1 - PTR_j y_{ij} \right] \prod_{s \neq j,\, s=1}^{k} 1 - PFR_s y_{is} \; + w_T \sum_{i=1}^{n} T_i$$

Objective Function:
First part: The penalty for accepting an undesired container
Second part: total tardiness

w_T	weight of tardiness in objective function
π_{ij}	probability of risk category j being present in container i
PFR_j	probability of false reject at station j
PTR_j	probability of true reject at station j
y_{ij}	1 if container i is inspected by station j, 0 o/w
T_i	tardiness of container i

$$\text{s.t.} \quad s_{ij} + t_{ij} y_{ij} - d_i - T_i \leq 0 \quad \forall i = 1,2,\ldots n; \forall j = 1,2,\ldots k$$

This constraint ensures that the completion time of any container does not exceed the due time plus tardiness of that container

s_{ij}	start time of inspection of container i at station j
t_{ij}	time inspection takes for container i at station j
d_i	due time for container i
y_{ij}	1 if container i is inspected by station j, 0 o/w
T_i	tardiness for container i

$$s_{ij} + t_{ij} y_{ij} - M \; 1 - z_{igj} \; - M \; 1 - y_{gj} \; - s_{gj} \leq 0 \quad \forall i \neq g, \forall j = 1,2,...k$$

$$s_{ij} + t_{ij} y_{ij} - M \; 1 - zz_{jhi} \; - M \; 1 - y_{ih} \; - s_{ih} \leq 0 \quad \forall i = 1,2,...n, \forall j \neq h$$

First constraint ensures that a station does not inspect two containers simultaneously.
Second constraint ensures that a container is not inspected by two inspection stations simultaneously.

s_{ij} ,s_{gj}	start time of inspection of container i,g at station j
s_{ih}	start time of inspection of container i at station h
t_{ij}	time inspection takes for container i at station j
y_{gj} $y_{ih.}$	1 if container i is inspected by station j, 0 o/w
zz_{igj}	1 if container i precedes container g on station j
d_i	due time for container I
T_i	tardiness for container i

Last 2 constraints are interesting…..

- They generate a large number of constraints because it considers all possible sequences of stations processing containers and containers visiting stations
- For example: $y_{gj} = y_{ij} = 1$
  - If *i* precedes *g* on station *j*, the following constraint must be met: $s_{ij} + t_{ij} \leq s_{gj}$
  - If g precedes i on station j, the following constraint must be met: $s_{ig} + t_{gj} \leq s_{ij}$
- For example: $y_{ij} = y_{ih} = 1$
  - If *i* is inspected on station *j* before station *h*, the following constraint must be met: $s_{ij} + t_{ij} \leq s_{ih}$
  - if *i* visits station *h* before station *j*, the following constraint must be met: $s_{ih} + t_{ih} \leq s_{ij}$

$$s_{ij} + t_{ij} y_{ij} - M \;\; 1 - z_{igj} \;\; - M \;\; 1 - y_{gj} \;\; - s_{gj} \le 0 \quad \forall i \ne g, \forall j = 1, 2, \ldots k$$

$$s_{ij} + t_{ij} y_{ij} - M \;\; 1 - zz_{jhi} \;\; - M \;\; 1 - y_{ih} \;\; - s_{ih} \le 0 \quad \forall i = 1, 2, \ldots n, \forall j \ne h$$

- To accommodate for the disjunctive nature of these required constraints, M (a large integer value) is multiplied by the “preceding” functions.

$$\min \quad 1-w_T \sum_{i=1}^{n}\sum_{j=1}^{k}\pi_{ij}\left[1-PTR_j y_{ij}\right]\prod_{s\neq j,\, s=1}^{k} 1-PFR_s y_{is} + w_T \sum_{i=1}^{n} T_i$$

$$\text{s.t.} \quad s_{ij} + t_{ij} y_{ij} - d_i - T_i \leq 0 \quad \forall i = 1,2,\ldots n; \forall j = 1,2,\ldots k$$

$$s_{ij} + t_{ij} y_{ij} - M\ 1 - z_{igj}\ - M\ 1 - y_{gj}\ - s_{gj} \leq 0 \quad \forall i \neq g, \forall j = 1,2,\ldots k$$

$$s_{ij} + t_{ij} y_{ij} - M\ 1 - zz_{jhi}\ - M\ 1 - y_{ih}\ - s_{ih} \leq 0 \quad \forall i = 1,2,\ldots n, \forall j \neq h$$

Where:

$$y_{ij},\ z_{igj},\ zz_{jhi} = 0 \text{ or } 1$$

$$s_i,\ T_i \geq 0$$

Now that we have an objective function with constraints, can we solve?

- Finding an optimal solution is difficult if not impossible when the number of inspection stations or containers is large.

- 8 weeks is not enough time to solve this.

←Sad face.

Lower Bound Function

- Given an inspection allocation *y*, we define a theoretical lower bound on the tardiness for an optimal schedule of *y*.
- Application of this lower bound within an algorithm allows partial solutions to the full problem to be compared for optimality, thereby increasing algorithm efficiency

In terms of individual containers:

- Each container must complete all inspections associated with allocation y, therefore: $$C_i \geq \sum_{j=1}^{k} y_{ij} t_{ij}$$
- If the total time of inspections of a container exceeds the due time of the container, the container experiences some positive tardiness

Hence

$$LB_1 = \sum_{i=1}^{n} \max\left(0, \sum_{j=1}^{k} y_{ij} t_{ij} - d_i\right)$$

Future work, and current progress

- Currently, I am looking at lower bound in terms of just inspection stations. Algorithm needs to be created.
- Special case when all times are equal, the lower bound can be tightened.
- The objective function w/ constraints can be solved using different methods: greedy heuristic.

THANKS!