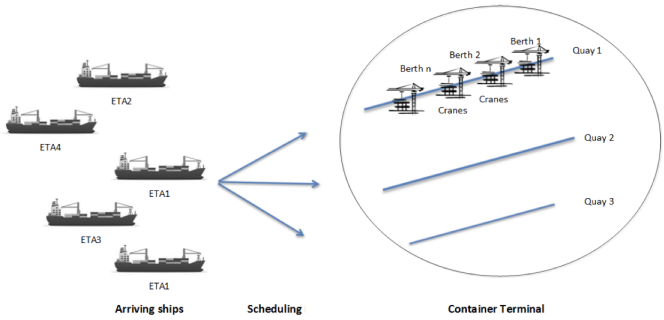
TÉCNICO LISBOA



IASD 2024/25
Project Assignment #1:
Berth Allocation Problem

Luis Custódio and Rodrigo Ventura
Instituto Superior Técnico, University of Lisbon

(Version 1.0, July 30, 2024)

1 Introduction

Over the last years, many companies have adopted the strategy of offshoring their manufacturing processes. This involves moving production facilities or supply bases to foreign countries with the goal of reducing production costs. This approach has gained significant popularity and led to a surge in the need for containerized marine transport and worldwide commerce. As a result, operators of marine container terminals (MCTs) are tasked with improving MCT operations, utilizing the resources at hand, to boost MCT throughput and efficiently meet the escalating demand.

For instance, the Port of Singapore holds a significant position in the global maritime industry because i) it is strategically located on the southern tip of the Malay Peninsula, making it a key stopover point for ships traveling between the Indian Ocean and the Pacific

Ocean; ii) it is the world's largest bunkering port, critical for importing natural resources and re-exporting products after they have been domestically refined; iii) since 1986, Singapore has been one of the world's busiest ports, with more than 130,000 vessels stopping by every year; iv) it is the focal point for some 200 shipping lines with links to more than 600 ports worldwide; and v) the maritime industry accounts for 7% of Singapore's GDP and provides 170,000 jobs. These factors collectively contribute to the importance of the Port of Singapore in the global maritime industry. [Source: Copilot]

Figure 1: Singapore Port (Source: MarineInsight)

Managing berth space for arriving vessels is a significant challenge in ports such as Singapore and Hong Kong. Given the limited availability of berth space and the high demand from numerous vessels, efficient management becomes critical to the smooth operation of these ports.

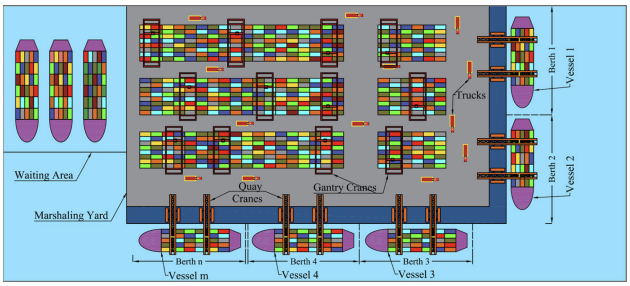


Figure 2: Marine Container Terminal (Source: Kavoosi et.al.)

The Berth Allocation Problem (BAP) is precisely the problem of allocating berth space for vessels in container terminals. A typical berth at the container terminal can accommodate multiple vessels at the same time. When no berth space is available, a vessel needs to wait for mooring.

This year's IASD project consists in developing a program using Python to solve Berth Allocation Problems. The project will be divided in three parts/assignments, each one with a deliverable. This first assignment aims at reading a BAP problem from an input file and determining the vessels' total processing time.

2 Problem Statement

In this project, we assume that time and space are discretize into integer units. The berth space has S sections indexed by $0, 1, 2, \ldots, S-1$, and there are N vessels, indexed by $0, 1, 2, \ldots, N-1$.

For each vessel i it is defined

a_i = arrival time of vessel i,
p_i = processing time for vessel i,
s_i = size of vessel i, measured in the number of berth sections, and $0 \leq s_i \leq S$,
w_i = weight (importance) of vessel i,

The following assumptions are made:

1. There is just one quay available.
2. Once a vessel is moored, it will remain in its location until all the required container processing is done.
3. Vessel processing times and vessel sizes are agreeable in general[1], meaning that if $s_i \geq s_j$, then $p_i \geq p_j$, and vice versa. This assumption reflects that a larger vessel requires a longer processing time.
4. A berth section handles at most one vessel at a time.

Thus, considering also the following variables for each vessel,

u_i = the starting mooring (processing) time for vessel i,
v_i = the starting berth section occupied by vessel i,
c_i = the departure time of vessel i,

if a vessel i is mooring at berth section v_i and time u_i, then the vessel occupies the consecutive berth sections between v_i and $v_i + s_i - 1$ and from time units u_i to $u_i + p_i - 1$, and $c_i = u_i + p_i$.

3 Objective

The goal of the project is to allocate berth space to vessels and to schedule the vessels such that the total weighted flow time is minimized. The flow time (f_i) of a vessel is the sum of the waiting time and the processing time, and can be computed as $f_i = c_i - a_i$.

The total weighted flow time is the sum of all flow times, each one multiplied by the vessel weight, where the weights reflect the relative importance of vessels, i.e.

[1]BA problems commonly operate under this assumption. However, it is important to note that several test files provided for this assignment do not adhere to this standard.

$$F = \sum_{i=1}^{N} w_i f_i \tag{1}$$

The final product of the project is a program developed using Python (version 3) capable of searching for an optimal solution for a BAP. The project will be divided in three parts, and three deliverables. This document defines the first assignment and deliverable.

Notice that problems may have more than one optimal solution, and the size of the problems vary.

4 First Assignment

The first assignment is to develop a Python program capable of (i) reading a BAP from an input file (format specified in the next section), (ii) calculating the cost of a given solution according to Equation (1), and (iii) checking if the solution complies with the constraints.

5 Input format (dat file)

The input files containing the initial configuration of a BAP are of type `.dat`. Each contains a single BAP configuration and is structured as follows:

- The input file may start with one or more comment lines. If a line starts with a '#' is considered a comment and be ignored;
- the first line, after any comment lines, has two integer numbers, defining the parameter S, the size of the berth space, and the parameter N, the number of vessels;
- then the following N lines in the file define the information for each vessel;
- each of these N lines has 4 integer numbers separated by a single space, defining a_i, p_i, s_i, w_i, respectively, for vessel i;
- the file ends with a blank line.

5.1 Input file: examples

In this section some files are presented as examples of input files for this assignment.

Example 1:

```
# this is a first comment line
# this is a second comment
12 4
10 3 2 1
```

```
11 3 2 1
3 9 5 1
0 5 4 1
```

This example involves a berth space of size 12 and 4 vessels. The first vessel arrives at time 10, has a processing (mooring) of 3 time units, has a size of 2 (i.e., needs 2 berth sections) to be moored, and has an importance (weight) of 1. The last (fourth) vessel arrives at time 0, has a processing of 5 time units, a size of 4, and a weight of 1.

Example 2:

```
# single comment line
5 3
0 3 3 1
0 3 3 1
0 3 3 2
```

Example 3:

```
5 3
0 3 3 1
3 3 3 1
5 3 3 6
```

6 Output representation

The solution to the BAP problem is a schedule for the N vessels, defining u_i, the starting mooring time, and v_i, the starting berth section, for each vessel i. The solution schedule is represented by a Python list, with N tuples, each with two items (u_i, v_i), in the same order as the vessels are defined in the input file.

Here are examples of schedule for the problems given in section 5.1

Example 1: [(10,0),(11,2),(3,4),(0,0)]

Example 2: [(3,0),(6,0),(0,0)]

Example 3: [(0,0),(8,0),(5,0)]

7 Deliverable

The Python program to be developed must (1) read an input file, and (2) determine the cost of a given solution.

The Python program to be delivered should be called `solution.py` and include (at least) a class with name `BAProblem` containing (at least) the following methods (see Annex A for the class template):

`load(fh)` loads a BAP from a file object fh

`cost(sol)` calculates the cost of the solution given as argument *sol*, where *sol* is a schedule, as defined in section 6.

`check(sol)` verifies if the solution given as argument *sol* is a feasible solution, i.e., it satisfies the problem constraints[2].

For instance, for solutions given in section 6 the cost F is 20, 21, and 29, respectively, and all three are feasible solutions.

For example 3, another possible solution is [(0,0),(3,0),(6,0)] with $F = 30$. Any other vessel permutation is worse than this two. So, the schedule [(0,0),(8,0),(5,0)] with $F = 29$ is the optimal solution for example 3.

8 Evaluation

The deliverable for this assignment is shown through DEEC Moodle, with the submission of a single python file, called `solution.py`, implementing the modules mentioned above. Instructions for this platform are available on the course webpage. Finally, the grade is computed in the following way:

- 50% from the public tests;
- 50% from the private tests; and
- -10% from the code structure, quality and readability.

Deadline: **27-September-2024**. Projects submitted after the deadline will not be considered for evaluation.

[2]returns True if yes; otherwise returns False.

Closing Remarks on Ethics:

- Any kind of sharing code outside your group is considered plagiarism;
- Developing your code in any open software development tool is considered sharing code;
- You can use GitHub. Make sure you have private projects and remove them afterward;
- If you get caught in any plagiarism, either by copying the code/ideas or sharing them with others, you will not be graded; and
- The scripts and other supporting materials produced by the instructors cannot be made public!

A Class Template

```
import search

class BAProblem(search.Problem):

        def __init__(self):
                """Method that instantiate your class.
                     You can change the content of this.
                     self.initial is where the initial state of
                     the BAP should be saved."""
                self.initial = None

        def load(self, fh):
                """loads a BAP problem from the file object fh.
                     You may initialize self.initial here."""
                pass

        def cost(self, sol):
                """Compute cost of solution sol."""
                pass

        def check(self, sol):
                """Check if solution sol satisfies problem constraints."""
                pass
```