# ALC 2021/2022
# $3^{rd}$ Project – Warehouse Packaging Scheduling with ASP

22/Nov/2021, version 1.0

## Overview

The 3rd ALC project is to develop a software tool for solving the Warehouse Packaging Scheduling problem. In order to solve this problem, students must use solvers for Answer Set Programming (ASP).

## Problem Specification

In many online shopping companies, products are stored in warehouses and there is a separate area for packaging of products according to the client's orders.

Consider the case of a warehouse where they have several people that are *runners*. The mission of a runner is to pick up products from shelves and put them into conveyor belts. The conveyor belts take the products to the packaging area of the warehouse.

Runners in warehouses are always on the move. Hence, they are not allowed to take breaks (pauses). The timespan of a runner is the time spent from the start until the runner puts the last product in the conveyor belt. In order to be have a sense of justice between runners, a runner cannot spend less than 50% of the maximum timespan of the other runners.

Let $P$ define a set of $m$ products located inside the warehouse in a given shelf. The warehouse operations needs to satisfy a set of $n$ orders $\{O_1, O_2 \ldots O_n\}$. Each order $O_j$ is composed of $k_j$ products, i.e. $O_j = \{p_1^j, p_2^j \ldots p_{k_j}^j\}$ such that each product $p_i^j \in P$, $1 \leq i \leq k_j$. In order to satisfy each order, a runner needs to go to the shelf where the product is located and put it into the conveyor belt. This must be done for all products in the order, not necessarily by the same runner.

Let $t_{ij}$ be the time a runner takes to go from the location of product $p_i$ to the location of product $p_j$ and put product $p_j$ in the conveyor belt. [1] **You can assume that the time between locations satisfies the triangular inequality, i.e.** $t_{ij} < t_{im} + t_{mj}$ **for any** $m \in P$

Let $c_i$ be the time product $p_i$ takes in the conveyor belt from its location in the warehouse to the packaging area. Observe that two products cannot arrive at the same time to the packaging

[1]Note that $t_{ii} = 1$ for all $1 \leq i \leq m$ since $t_{ii}$ only includes the time to add the product to the conveyor belt.

| $t_{ij}$ | $p_1$ | $p_2$ | $p_3$ | $p_4$ |
|---|---|---|---|---|
| $p_1$ | 1 | 5 | 3 | 3 |
| $p_2$ | - | 1 | 3 | 3 |
| $p_3$ | - | - | 1 | 2 |
| $p_4$ | - | - | - | 1 |

Table 1: Runner's time between product shelves.

|  | $p_1$ | $p_2$ | $p_3$ | $p_4$ |
|---|---|---|---|---|
| $c_i$ | 3 | 1 | 3 | 2 |

Table 2: Time in conveyor belt for each product.

area. At each time step, you can only have one product arriving to the packaging area in order to allow a proper processing of the product and avoid blockages.

Considering a fixed number of runners, the goal is to determine the sequence of products each runner should put in the conveyor belt such that you minimize the overall timespan of the operations, i.e. the timespan from the initial timestep until the last product arrives at the packaging area.

Consider a problem instance with two runners in the warehouse and that two orders $O_1$ and $O_2$ such that $O_1 = \{p_1, p_2, p_3\}$ and $O_2 = \{p_1, p_4\}$ are to be satisfied. Furthermore, table 1 defines the time a runner takes between product locations in the warehouse and table 2 defines the time a product takes from its location to the packaging area.

Suppose the two runners are initially both located in $p_1$. In this case, the optimal solution would be as shown in Figure 1. For this example, the optimal solution has a cost of 8 since this is the timestamp that the last product arrives at the packaging area.

Note that all products arrive at the packaging area at different timestamps. No two products can arrive at the same time. Moreover, runners are always on the move with no pauses. Finally, the maximum timespan of the runners is 7 (Runner 1) and all runners have a timespan of at least 50% of the maximum.

## Project Goals

You are to implement a tool, or optionally a set of tools, invoked with command `proj3`. This set of tools **must** use an ASP solver to compute the warehouse packaging scheduling problem.

**Your tool does not take any command-line arguments. The problem instance is to be read from the standard input.**

Consider an instance file named `job.wps`. The tool is expected to be executed as follows:

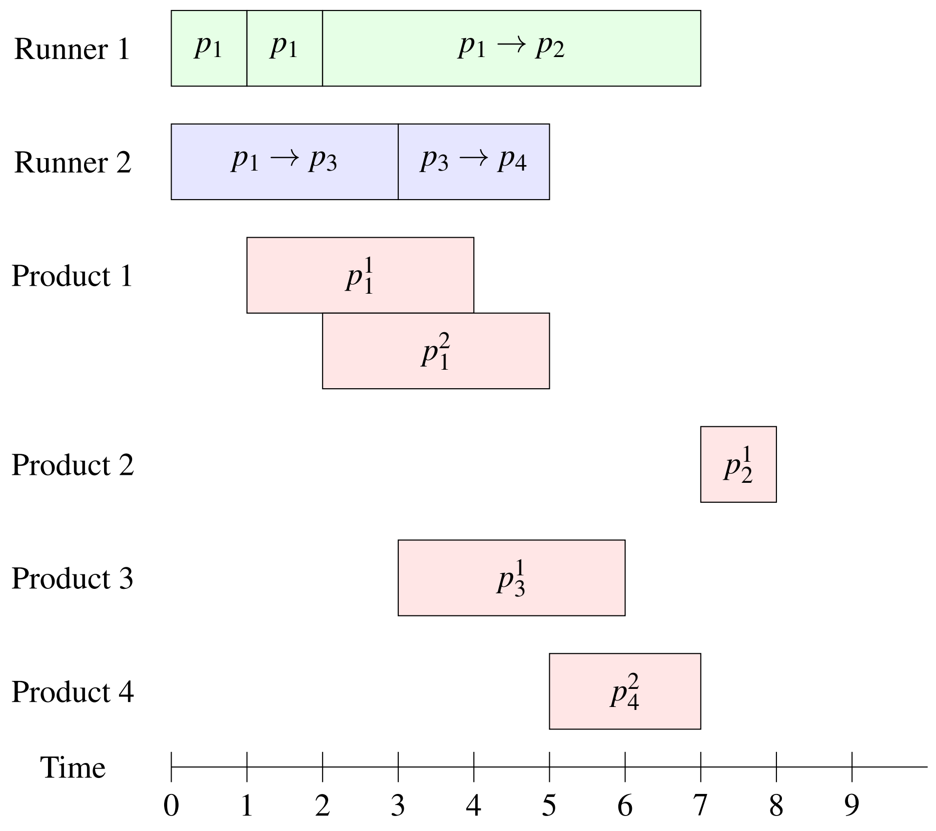

Figure 1: Warehouse Packaging Scheduling Solution

```
proj3 < job.wps > solution.txt
```

The tool must write the solution to the standard output, which can then be redirected to a file (e.g., `solution.txt`).

The programming languages to be used are only C/C++, Java or Python. The formats of the files used by the tool are described below.

## File Formats

You can assume that all input files follow the description provided in this document. There is no need to check if the input file is correct. Additionally, all lines (input or output) must terminate with the end-of-line character.

## Input Format

The input file representing a problem instance is a text file that follows the following format:

- One line with an integer $n (n > 1)$ defining the number of runners
- One line with an integer $m (m \geq n)$ defining the number of products
- One line with $n$ integers defining the initial position of each runner
- A sequence of $m$ lines where the $i^{th}$ line contains $m$ integers describing the times $t_{ij}$ to go from the location of product $i$ to the location of product $j$.
- One line with $m$ integers where the $i^{th}$ integer defines the time product $i$ takes in the conveyor belt to arrive to the packaging area.
- One line with an integer $o (o > 1)$ defining the number of orders
- A sequence of $o$ lines where the $i^{th}$ line describes the products of order $i$. Each line contains the following:
  - An integer $k_i (k_i > 0)$ defining the number of products in order $i$
  - A sequence of $k_i$ integers denoting the products in order $i$

Finally, observe that products are always numbered from 1 to $m$ and that integers in the same line are always separated by a white space.

## Output Format

Considering the several constraints a solution must satisfy, it is possible that some problem instances do not have a solution. If the problem instance is not satisfiable, then the output of the program is a single line with the word UNSAT. Otherwise, the output representing an optimal solution to the problem instance must comply with the following format:

- One line with an integer $T$ defining the optimal timespan for the warehouse scheduling problem
- A sequence of $n$ lines where the $i^{th}$ line defines the sequence of products handled by runner $i$ as follows:
  - An integer $k_i$ denoting the number of products of runner $i$
  - A sequence of $k_i$ integers denoting the sequence of products handled by the runner
- A sequence of $o$ lines where the $j^{th}$ line defines the timestamp that each product of order $O_j$ is put on the conveyor belt. Each line is defined as follows:
  - An integer $k_j$ denoting the number of products of order $O_j$
  - A sequence of pairs of $k_j$ integers where each pair contains the product identifier and the respective timestamp that the product is put on the conveyor belt. The product identifier and the timestamp are separated by the character $:$.

**Important:** The final version to be submitted for evaluation must comply with the described output. Project submissions that do not comply will be severely penalized, since each incorrect output will be considered as a wrong answer. An application that verifies if the output complies with the description will be available on the course's website.

## Example 1

The file describing the problem in Table 1 is as follows:

```
2
4
1 1
1 5 3 3
5 1 3 3
3 3 1 2
3 3 2 1
3 1 3 2
2
3 1 2 3
2 4 1
```

The optimal solution corresponding to Figure 1 would be:

```
8
3 1 1 2
2 3 4
3 1:1 2:7 3:3
2 1:2 4:5
```

## Example 2

Consider the following problem instance with 2 runners starting in position of product 1, 2 products and 2 orders as follows:

```
2
2
1 1
1 10
10 1
2 3
```

```
2
2 1 2
1 1
```

In this case, it is not possible to satisfy the constraint that the timespan of each runner is at least 50% of any other runner. Hence, the output would be:

```
UNSAT
```

## Additional Information

The project is to be implemented in groups of one or two students.

The project is to be submitted through the course website. Jointly with your code, you should submit a short text file describing the main features of your project.

The evaluation will be made taking into account correctness given a reasonable amount of CPU time (80%) and efficiency (20%).

The input and output formats described in this document must be strictly followed.

## Project Dates

- Project published: 22/11/2021.
- Project due: 31/01/2022 at 23:59.

## Omissions & Errors

Any detected omissions or errors will be added to future versions of this document. Any required clarifications will be made available through the course's official website.

## Versions

22/Nov/2021, version 1.0: Original version.

## References