Challenge ROADEF / EURO 2018
Cutting Optimization
Rules

Lydia Tlilane and Quentin Viaud
Saint-Gobain, Datalab

July 9, 2018

Contents

1 General rules **2**

2 Organization **2**
2.1 Datasets . . . 3
2.2 Evaluation . . . 3
2.3 Ranking function . . . 3
2.4 Computing and software limitations . . . 5

3 Files format and description **6**
3.1 Input files . . . 6
3.1.1 Batch file . . . 6
3.1.2 Defects file . . . 7
3.1.3 Global parameter file . . . 8
3.2 Output file . . . 8
3.3 Requirements for output files . . . 9
3.4 Solution checker and visualization tool . . . 10

4 Discussion and updates **10**

5 Intellectual property **11**

1 General rules

The ROADEF/EURO 2018 challenge is dedicated to cutting problems in collaboration with Saint-Gobain Glass France (SGGF). The problem description, instances and checker are proposed by SGGF. Participants compete to propose the best algorithmic solutions to solve the given problem. Informations about the challenge are available on the challenge website. The participants are welcomed to provide feedbacks (questions, doubts, errors, ...) to the organizers concerning the problem. They must submit them on the forum available in the challenge website. Questions asked by email might be ignored by the organization. The organizers try to answer to the participants as fast and accurately as possible.

The rules about team composition are given below:

- A team can be composed of any number of persons
- One person cannot be a member of more than one team
- Each team can submit only one program to solve the problem
- A junior team must include only student members (students enrolled in a Licence, Master's or Doctorate program).
- A team is not considered as a junior team if one or more of its members has defended a Ph.D. on December 31st, 2017.
- Persons employed by Saint-Gobain Group cannot participate
- The members of the organization committee cannot participate, but members of their respective laboratories can. In such cases, the organizers do not share with them any information related to the challenge unless it has been previously communicated to all participants.
- Anyone having signed a contract with Saint-Gobain Glass France and/or Saint-Gobain Recherche to work on topics directly related to the subject of the challenge cannot participate.
- To participate in open source category, source code must be available on an open platform (GitHub,...) at the end of the competition.

2 Organization

This section describes the general steps of the challenge.

2.1 Datasets

Three datasets composed of industrial problem instances are provided to participants during the challenge:

- Dataset A of 20 instances at the launch of the challenge.
- Dataset B of 15 instances available after the end of the qualification phase.
- Dataset X of 15 hidden instances that will be used to rank the candidates for the final phase and available after the end of the challenge.

Dataset A contains small/medium size instances while datasets B and X contain regular size instances. An item batch is composed of at most 700 items. The number of stacks is bounded by the number of items, *i.e.* at most 700 stacks. The number of available bins in all instances is always 100 which allows to find a feasible solution. A bin contains at most 8 defects, so there are at most 800 defects per instance of the problem.

2.2 Evaluation

During the sprint phase, teams are evaluated according to their submitted solution files. At the deadline of the qualification and final phases, each team must send its computer program (see Section 2.4), solution files and a short document (2 pages max. for the qualification and 5 pages max. for the final phase). This document must describe the rationale of the solving method, the characteristics of the computer used to find solutions and the computational results table. The dataset A will be used to compare teams during the sprint phase and at the end of the qualification phase, datasets B and X at the end of the final phase.

The computer program of each team will be executed with a time limit of three minutes and another one of one hour. Let Z be a problem instance from a dataset and the solution value for instance Z within three minutes (resp. one hour) is $V_3(Z)$ (resp. $V_{60}(Z)$). The following function is used to compute the score $V(Z)$ of a team for instance Z:

$$V(Z) = 0.1V_3(Z) + 0.9V_{60}(Z)$$

The weight function between the two solutions is favorable to the one hour time limit since it is the most frequent used case. Note that if a team fails to provide a solution within one of the time limits for an instance Z or if the retained solution is considered as not valid by the solution checker, the result of a very naive greedy heuristic will be used in the score computation of the team.

2.3 Ranking function

At the end of each phase, teams are ranked according to their solution values. The used ranking function is the following.

Each team earns points for each instance, depending on their result. We denote by score the number of points earned by a team, whereas result denote the objective value obtained by the team. The winner is the team that maximizes the sum of its scores over all instances.

Let m be the total number of instances in a dataset. Let y_{ij} be the result of team i for instance j computed according to the evaluation function from Section 2.2. Let $n_{better(ij)}$ be the number of teams with result strictly better than the result of team i on instance j :

$$n_{better(ij)} = \text{Card}(\{k \neq i \text{ s.t. } y_{kj} < y_{ij}\}).$$

Let R be the maximal score that a team can earn from one instance. For the qualification phase, $R = 5$ over all instances of dataset A. For the final phase, $R = 10$ over all instances of datasets B and X.

Let p_{ij} be the number of points scored (i.e. the score) by team i for the instance j. It is defined by

$$p_{ij} = \begin{cases} \max\{0, R - n_{better(ij)}\} & \text{if the solution feasible,} \\ 0 & \text{if the solution is unfeasible or the program crashes.} \end{cases} \tag{1}$$

The global score of a team, denoted as $score(i)$ is defined by :

$$score(i) = \sum_{j \in [1,m]} p_{ij}.$$

In final phase, this ranking function will be applied separately for junior teams, senior (non-junior) teams and for the open source category.

The following illustrates how the top team is identified. Consider the following example with $m = 6$ instances.

Instance	1	2	3	4	5	6
Team 1	34	35	42	32	10	12
Team 2	32	24	44	33	13	15
Team 3	33	36	30	12	10	17
Team 4	36	32	46	32	12	13
Team 5	37	30	43	29	9	4
Team 6	68	29	41	55	10	5
Team 7	39	30	43	58	10	4

Table 1: Values of y_{ij}

The values $n_{better(ij)}$ are the following:

Instance	1	2	3	4	5	6
Team 1	2	5	2	2	1	3
Team 2	0	0	5	4	6	5
Team 3	1	6	0	0	1	6
Team 4	3	4	6	2	5	4
Team 5	4	2	3	1	0	0
Team 6	6	1	1	5	1	2
Team 7	5	2	3	6	1	0

Table 2: Values of $n_{better(ij)}$

The scores are the following:

Instance	1	2	3	4	5	6
Team 1	3	0	3	3	4	2
Team 2	5	5	0	1	0	0
Team 3	4	0	5	5	4	0
Team 4	2	1	0	3	0	1
Team 5	1	3	2	4	5	5
Team 6	0	4	4	0	4	3
Team 7	0	3	2	0	4	5

Table 3: Values of score p_{ij}, with $R = 5$

Finally, the global scores are:

Team 1	15
Team 2	11
Team 3	18
Team 4	7
Team 5	20
Team 6	15
Team 7	14

Table 4: Values of $score(i)$, with $R = 5$

In this example, Team 5 would be the winner.

2.4 Computing and software limitations

To make a fair comparison of methods developed by different participating teams, each of them has to deliver for qualification and final phases an executable code. This program must take as input a problem instance from one of the datasets according to format from Section 3.1. It must return an output solution file using standards defined in Section 3.2. For each instance, the organizers will run ONE trial with fixed seed chosen at random. The same seed

will be used for each team and each instance. Using the seed value passed as a parameter to the executable program is under the responsibility of participants to support the repeatability of experiments/evaluations, particularly, if their solution methods are based on probabilistic frameworks or components. Programs will be evaluated only ONCE, never TWICE. If variability occurs, the organizers can not be responsible from potential bad behavior on this single run. The team's submissions will be run by the organizers, they should thus give support to the organizers in the process of running their algorithms.

The computer that should be used to evaluate programs of candidates is a Linux OS machine (Ubuntu 18.04) with 4 CPU (8 Threads) with 3.6GHz - Intel® Xeon® W-2123 Processor, 48GB of RAM and a graphic card of 5GB. It is allowed to use both CPUs and GPUs on the machine. The following list of ILP solvers is allowed during the challenge:

- CPLEX 12.8
- LocalSolver 8.0

If a team wants to use extra solvers or libraries, it has to be asked first to the members of the organization. If the answer is positive, participants must provide a description on how to use and install them. In that case, it will be better to do it before deadlines of the different challenge steps. You can use your favoured programming language to develop algorithms. However programming in C/C++ could facilitate potential collaborations at the end of the challenge.

3 Files format and description

This section describes files format used during the challenge.

3.1 Input files

The input files that contain all required data for an instance "id" are given in CSV format (comma-separated values), with ";" as separator and the first line always contains headers (name of columns). For example, in the dataset A, the first problem instances is composed of two files: $A1\_batch.csv$ and $A1\_defects.csv$. The first one stores the data about items, the second about defects on plates. Files $B10\_batch.csv$ and $B10\_defects.csv$ are related to the 10-th instance of dataset B. An extra file called $global\_param.csv$ has also to be considered. It contains the set of standard parameters to solve the problem. A more detailed description of each file and its structure is given hereinafter. Through the rest of this section a fictional instance $A0$ is used as example.

3.1.1 Batch file

The batch file contains a batch of items to be cut.

ITEM_ID	LENGTH_ITEM	WIDTH_ITEM	STACK	SEQUENCE
0	600	600	0	1
1	600	600	0	2
2	1242	247	0	3

Figure 1: Batch file for instance $A0$

Each row (except the header) corresponds to an item with the following columns:

- ITEM_ID : identifier of the item.
- LENGTH_ITEM: length of the item.
- WIDTH_ITEM : width of the item with LENGTH_ITEM $\geq$ WIDTH_ITEM.
- STACK: identifier of the stack in which ITEM_ID is assigned.
- SEQUENCE: sequence or order of ITEM_ID inside STACK (from 1 to |STACK|).

Note that it does not matter how the item dimensions are considered since their rotation is allowed.

3.1.2 Defects file

The defect file contains all the defects related to the available plates.

DEFECT_ID	PLATE_ID	X	Y	WIDTH	HEIGHT
0	0	50	50	3	3
1	0	3	3207	3	3
2	3	1090	1685	2	4

Figure 2: Defect file for instance $A0$

Each row is related to one defect with the following columns:

- DEFECT_ID: identifier of the defect.
- PLATE_ID: corresponds to the identifier of the plate (or bin) in which the defect is located.
- X: x coordinate of the bottom left corner of the defect in PLATE_ID.
- Y: y coordinate of the bottom left corner of the defect in PLATE_ID.
- WIDTH: width along the x-axis of the defect.
- HEIGHT: height along the y-axis of the defect.

3.1.3 Global parameter file

The file *global_param.csv* is common to all instances and contains some fixed optimization parameters or constraint values. The content of this file is summarized in the following table:

NAME	VALUE	COMMENTS
nPlates	100	Number of available plates (bins) that should not be exceeded
widthPlates	6000	Fixed width of plates
heightPlates	3210	Fixed height of plates
min1Cut	100	Minimum distance between two consecutive 1-cuts (except wastes)
max1Cut	3500	Maximum distance between two consecutive 1-cuts (except residual)
min2Cut	100	Minimum distance between two consecutive 2-cuts (except wastes)
minWaste	20	Minimum width and height of wastes

Table 5: Content of *global_param.csv* file

Note that it is not necessary to add an input file related to the plates due to their standard size. The identifiers of plates are always $0, 1, \ldots, nPlates - 1$ that must be used in that order.

3.2 Output file

The filename of the solution corresponding to the first instance in dataset A must be named *A1_solution.csv*. Respectively, for the 10-th instance of dataset B it must be *B10_solution.csv*. A solution is represented by a forest where each cutting pattern is represented by a tree in the forest.

PLATE_ID	NODE_ID	X	Y	WIDTH	HEIGHT	TYPE	CUT	PARENT
0	0	0	0	6000	3210	-2	0	
0	1	0	0	600	3210	-2	1	0
0	2	0	0	600	53	-1	2	1
0	3	0	53	600	600	0	2	1
0	4	0	653	600	600	1	2	1
0	5	0	1253	600	1242	-2	2	1
0	6	0	1253	247	1242	2	3	5
0	7	247	1253	353	1242	-1	3	5
0	8	0	2495	600	715	-1	2	1
0	9	600	0	5400	3210	-3	1	0

Figure 3: Overview of solution file for *A*0 instance

In the output file in Figure 3, each row must correspond to a node of a tree of the forest. A node is always a rectangle that may be a plate, a branch (node

with children), an item, a waste or a residual. The following informations should be given for each node in that order of columns:

- PLATE_ID: identifier of the plate that corresponds to the current cutting pattern.
- NODE_ID: each node must have his own identifier that you set.
- X: x coordinate of the bottom left corner of the node.
- Y: y coordinate of the bottom left corner of the node.
- WIDTH: width along the x-axis of the node.
- HEIGHT: height along the y-axis of the node.
- TYPE:

  ≥ 0 ITEM_ID.

  $= -1$ Waste.

  $= -2$ Branch (node with children).

  $= -3$ Residual.

- CUT:

  $= 0$ plate.

  $= \alpha$ α-cut, for $\alpha \in \{1, 2, 3, 4\}$.

- PARENT: NODE_ID of the parent node. If no parent, leave a blank cell.

3.3 Requirements for output files

- Reported data in output files have to be integer.
- When creating the output forest:
  - Always add the header as it is done in the example (see Figure 3).
  - The trees must be given in the correct sequence of plates, *i.e.* first the nodes of plate 0, then nodes of plate 1....
  - For each plate, the root (node with CUT=0) should be given first.
  - The children of a node should be given from left to right (as represented on the cutting pattern).
  - The nodes of trees should be given in depth first search.
  - If the last 1-cut of the forest in the last cutting pattern is a waste, it must be declared as a residual.

These requirements are necessary to use the checker, the visualization tool and to validate solutions.

3.4 Solution checker and visualization tool

A checker of solutions is available to test the feasibility of the produced solutions. The source code of the checker is also available in C++. The checker allows to:

- parse the input files and the solution of an instance
- check the feasibility of the solution
- compute the value of the objective function and some other data (number of plates used, size of residual, etc.)
- visualize the resulting cutting patterns on virtual plates. It requires a browser that supports HTML5.

The use of the checker and the visualization tool require to have input and output files in the formats described in Sections 3.1-3.2. Please refer to *readme.txt* available in the checker project for use. The Figure 4 shows the output produced by the checker for instance $A0$.

Figure 4: Visualization of a solution for instance $A0$

4 Discussion and updates

To help participants managing their project, the organizers try to communicate immediately changes that might occurs during the challenge. This includes clarification of the problem description or change in problem instances for example. The organizers try to avoid making such changes unless they find it necessary for the challenge.

The organizers reserve the right to disqualify any participant or team from the competition at any time if the participant is considered to have worked outside the spirit of the competition rules.

5 Intellectual property

1. Participants have intellectual property on their computer programs developed during the challenge. Saint-Gobain Glass France and any third party may use information provided by the participants through technical reports, scientific papers and oral presentations, but cannot use a computer program without the agreement of the team who wrote this program.

2. Participants to the challenge cannot claim to have a partnership or a contract with Saint-Gobain Glass France or any entities of the Saint-Gobain Group, even if they win the challenge. They can only claim to be participants (respectively qualified / winner) if it is the case.

3. Saint-Gobain Glass France may (but has taken no engagement to) sign contracts with some participants after the challenge. Any such contract would be independent of the challenge.