Combinatorial Decision Making and Optimization

Present wrapping problem
SMT solution

Filippo Orazi 0000928971
Andrea Rossolini 0000954735

February 2021

Contents

1 Introduction

The present wrapping problem is a combinatorial problem that focuses on fitting all the paper needed to wrap a certain amount of present in a bigger sheet of fixed dimension.
The instances we had to solve provided rectangular pieces to be cut out of a rectangular sheet, each piece and the sheet had fixed dimension specified in the input files.

2 Model

The SMT (Satisfiability Modulo Theories) solution used the python library z3py that integrates the z3 solver to the python environment.

2.1 main problem constraints

Given the bigger sheet as the domain (D) with its dimensions (D_{height}, D_{width}) and the dimension of each piece p_i (p_{i_w}, p_{i_h}) we want to find the coordinates (p_{i_x}, p_{i_y}) of the lower left corner of each piece. The basic constraints for the main problem are:

- All the lower left corners need to stay in the domain:

$$\begin{aligned} \forall\, p \in P : 0 < p_y < D_{height} \\ \wedge \\ 0 < p_x < D_{width} \end{aligned} \tag{1}$$

- All the pieces need to fit in the domain:

$$\begin{aligned} \forall\, p \in P : 0 < p_y + p_h < D_{height} \\ \wedge \\ 0 < p_x + p_w < D_{width} \end{aligned} \tag{2}$$

- Two pieces can not overlap:

$$\begin{aligned} \forall\, p_i, p_j \in P \wedge i \neq j : & \; p_{i_y} + p_{i_h} < p_{j_y} \\ & \wedge \\ & p_{j_y} + p_{j_h} < p_{i_y} \\ & \wedge \\ & p_{i_x} + p_{i_w} < p_{j_x} \\ & \wedge \\ & p_{j_x} + p_{j_w} < p_{i_x} \end{aligned} \tag{3}$$

2.2 Implied constraints

The implied constraints assures that the sum of the vertical dimension of all pieces in a certain row is not greater than the width of the domain

$$\forall\, y \in D\ :(\sum_{i=1}^{N} p_{i_w} \cdot [p_{i_h} + p_{i_y} > y \wedge p_{i_y} < y]) < D_{width} \tag{4}$$

Where y represents a column in the domain D and N is the number of pieces. Similarly for the height:

$$\forall\, x \in D\ :(\sum_{i=1}^{N} p_{i_h} \cdot [p_{i_w} + p_{i_x} > x \wedge p_{i_x} < x]) < D_{height} \tag{5}$$

In these formula the condition $[p_{i_w} + p_{i_w} > y \wedge p_{i_x} < x]$ finds out if a certain piece is set in a column by checking if the column is between the lower left angle and the lower right angle. Using Iverson notation [1] the condition inside the squared brackets is evaluated to 0 if it's $false$ and 1 if it's true.

2.3 Global constraints in CP

Not applicable to the SMT solution

2.4 Best way to search for solution in CP

Not applicable to the SMT solution

2.5 Rotation

In the SMT program the rotation has been expressed as a possible switch between the two available dimension in the input. We indicate as $dim_1(p), dim_2(p)$ the two number that can be found in the input representing the piece p. Previously $dim_1(p)$ was considered to be p's width while $dim_2(p)$ was it's height ($p_w = dim_1(p)$ and $p_h = dim_2(p)$). When considering rotation we concluded that a 90° and 270° rotation can be simulated by inverting the two dimensions. We then put the dimension of each piece (p_h, p_w as a variable to be found. In particular:

$$\forall\, p \in P \text{ where } \ dim_1(p) \neq dim_2(p) :$$
$$(p_w = dim_1(p) \wedge p_h = dim_2(p)) \vee (p_w = dim_2(p) \wedge p_h = dim_1(p))$$

With this constraint is the solver that decides which is the width and which the height depending on its necessities. The condition $dim_1(p) \neq dim_2(p)$ removes possible symmetries related to pieces with same height and width that don't change when rotated:

$$\forall\, p \in P\ where\, dim_1(p) = dim_2(p) :$$
$$(p_w = dim_1(p) \wedge p_h = dim_2(p))$$

For those pieces we arbitrarily set $p_w = dim_1(p)$ and $p_h = dim_2(p)$.

2.6 Same dimension pieces

This symmetry has been dealt with using a new constraint. For each two pieces that have the same dimensions (also considering rotation) we force them into a fixed relative position where the second piece cannot be on the lower-left side of the first:

$$\forall\, p_i, p_j \in P \; where$$
$$(dim_1(p_i) = dim_1(p_j) \wedge dim_2(p_i) = dim_2(p_j))$$
$$\vee$$
$$(dim_1(p_i) = dim_2(p_j) \wedge dim_2(p_1) = dim_1(p_j)) :$$
$$p_{i_x} \leq p_{j_x} \wedge ((p_{i_x} = p_{j_x}) \implies (p_{i_y} < p_{j_y})) \tag{6}$$

Notice that in the SMT the condition is checked by the python code and thus doesn't influence the final computation effort.

3 Results

3.1 SMT

All the result are computed using the z3 library for python. Moreover, all the results that are not reported in the tables are the tests that take more than 1000 seconds to converge. Some Results are reported even if their execution time is over the limit as an example.
Table 1 shows the average time and the standard deviation of each input. When the std is very high it could signify that the search highly depends on which heuristic z3 chooses for that particular execution.

Paper size	Static		Rotation	
	time	**std**	**time**	**std**
8x8	0.0096 s	0.0008	0.0268 s	0.0230
9x9	0.0100 s	0.0007	0.0295 s	0.0166
10x10	0.0398 s	0.0535	0.0380 s	0.0268
11x11	0.0544 s	0.0693	0.1635 s	0.0077
12x12	0.0625 s	0.0548	0.5530 s	0.1866
13x13	0.0860 s	0.0725	5.1495 s	0.4787
14x14	0.0920 s	0.1008	2.4930 s	0.4822
15x15	0.0907 s	0.1232	0.1180 s	0.0084
16x16	0.1307 s	0.0736	7.4855 s	1.5365
17x17	0.2285 s	0.1982	7.1085 s	0.9722
18x18	6.7530 s	5.2967	204.2300 s	61.0204
19x19	1.4892 s	2.2999	302.8964 s	59.8304
20x20	1.2238 s	1.7781	10.4235 s	1.9905
21x21	0.7538 s	0.6576	6.2880 s	1.5231
22x22	2.6258 s	2.4036	15.0324 s	3.2222
23x23	28.3535 s	19.5626	3199.3590 s	-
24x24	11.0907 s	8.7278	633.5915 s	-
25x25	6.9873 s	4.4181	2464.8530 s	-
26x26	47.4773 s	20.4637	404.0700 s	-
27x27	26.6185 s	7.5182	1122.2880 s	-
28x28	29.8388 s	9.8257	3128.4450 s	-
29x29	281.9068 s	398.1212	-	-
30x30	22.8970 s	5.7692	-	-
31x31	10.1603 s	9.6978	-	-
32x32	99.9996 s	53.5026	-	-
33x33	29.5341 s	8.3691	-	-
34x34	11.4531 s	11.3202	-	-
35x35	42.1710 s	56.0233	-	-
36x36	45.6121 s	49.1566	-	-
37x37	490.6572 s	514.0379	-	-
38x38	10.4686 s	10.1954	-	-
39x39	10705.8972 s	3423.6804	-	-
40x40	8.6548 s	10.6190	-	-

Table 1: SMT average time and standard deviation over the tests. The "-" symbols means that either the execution was too expensive or that the variance couldn't be calculated due to too few experiments.

References

[1] D. E. Knuth. Two notes on notation.