Report for Project Work

Combinatorial Decision Making and Optimization

Module 1

(CP Solution)

Yuwei Ke (yuwei.ke@studio.unibo.it)

Yiran Zeng (yiran.zeng@studio.unibo.it)

Contents

1 Introduction

VLSI (Very Large Scale Integration) refers to the trend of integrating circuits into silicon chips. A typical example is the smartphone. The modern trend of shrinking transistor sizes, allowing engineers to fit more and more transistors into the same area of silicon, has pushed the integration of more and more functions of cellphone circuitry into a single silicon die (i.e. plate). This enabled the modern cellphone to mature into a powerful tool that shrank from the size of a large brick-sized unit to a device small enough to comfortably carry in a pocket or purse, with a video camera, touchscreen, and other advanced features.In this project,a fixed-width plate and a list of rectangular circuits are given,all the circuits should be put on the plate and the height need to be minimized.

2 Implementation

The code is implemented in python with minizinc library.All the input and output are done in python by using the constraint programming model defined in minizinc.The solver is Gecode 6.3.0.

2.1 parameters

2.1.1 Input

The input variables are the following ones:

w - the width of the silicon plate
n - the number of circuits
[$width_1$,$width_2$,..,$width_n$] - the width of every circuit
[$height_1$,$height_2$,..,$height_n$] - the height of every circuit

2.1.2 Output

2.1.2.1 Normal model output

Once the program finished optimization process and find solution,it will give output as following:

w h
n
$Width_1$ $height_1$ x_1 y_1
$Width_2$ $height_2$ x_2 y_2
.....
$Width_n$ $height_n$ x_n y_n

Among these parameters:

- h is the minimum height the program needs to optimize.
- x_i and y_i correspond to the left bottom point's x-coordinate and y-coordinate of the i circuit.
- w,n,$width_i$,$height_i$ are same as the input.

And it will generate a picture as Figure 1:

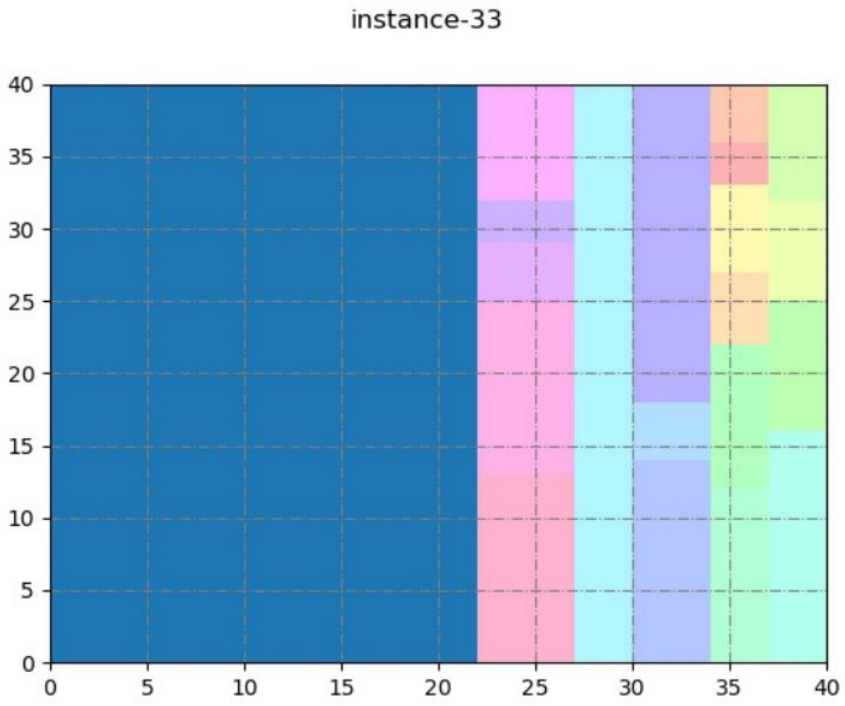


Figure 1:Instance 33 with width=40,n=20 and height=40

2.1.2.2 Rotation model output

Output format is as following:

w h
n
$Width_1$ $height_1$ x_1 y_1 $Rotation_1$
$Width_2$ $height_2$ x_2 y_2 $Rotation_2$
.....
$Width_n$ $height_n$ x_n y_n $Rotation_2$

- Rotation is boolean variable.When it is true,it means the circuit has been rotated.On the contrary,false means it is not rotated.
- Other variables' meaning is same as 2.1.2.1
- It will also generate a picture like Figure 1.

3 Normal model Constraints

3.1 Implied constraint

When taking in to account this problem,we can not ignore the implied constraint of the problem itself.As the plate's width is fixed,all the circuits inside could not exceed the left,right and button boundary of the plate.In the solution,the left-bottom corner of plate is the coordinate's origin,and bottom edge is X axis and left edge is Y axis.So there should have formulae:

$$0 < x_i < w \quad (0 < i <= n)$$

$$\max(x_1 + width_1, x_2 + width_2, .., x_n + width_n) <= w$$

$$\max(y_1 + \text{height}_1, y_2 + \text{height}_2, .., y_n + \text{height}_n) <= h$$

3.2 Global constraints

3.2.1 cumulative

In this strip packing problem,it can be treated as resource scheduling problem.Each circuit can be viewed as a task, the horizontal axis of the plate corresponds to the time, the vertical axis corresponds to the capacity, the start time and duration of each task can be viewed as the horizontal axis and width of the circuit, while the amount of resources required corresponds to the height.On the other hand,when the duration be represented by vertical axis and the amount of resource is showed by horizontal one,that is also reasonable.Thus,the global constraint cumulative() can be used at here:

$$\text{cumulative}(x, \text{width}, \text{height}, h)$$
$$\text{cumulative}(y, \text{height}, \text{width}, w)$$

3.2.2 No Overlapping

Circuit cannot overlap with other circuits, so some restrictions are required on the code. In the minizinc tutorial, there is a global constraint Diffn () which can realize the no overlapping function by giving circuits's origins and sizes:

$$\text{diffn}(x, y, \text{width}, \text{height})$$

3.2.3 Remove gap

To make sure there is no gap between circuits which means that each circuit is complete fit on another piece of circuit so we also need to add some other constraint.

$$(x_i = 0 \vee x_i = x_j + \text{width}_j)$$
$$\wedge$$
$$(y_i = 0 \vee y_i = y_j + \text{height}_j)$$

3.3 Symmetry breaking constraints

Symmetry may lead to a solution/failure which will have many symmetrically equivalent states.For example,exchange circuits with same width and height;flip the plate horizontally or vertically;rotate the plate 180°. In order to get higher efficiency,we need to reduce the solution and search space by defining some symmetry breaking constraints:

- Sort all circuits by area from largest to smallest,and let the one with the largest area in the lower part of the second largest circuit.In this way,the circumstances including horizontal and vertical flip as well as 180° rotation are all ruled out.

$$\text{let}\{\text{int: } o1 = \text{circuitOrder}[1], \text{int: } o2 = \text{circuitOrder}[2]\}$$
$$\text{in lex\_less}([y[o1], x[o1]], [y[o2], x[o2]])$$

- To limit the switching of circuits with same width and height,If the width and height of circuit i and circuit j are the same then if their y's are equal then x_j has to be greater than x_i and if they aren't then y_j has to be greater than y_i. For every pair of circuits:

```
forall(r in Circuit)(let {array[int] of int: R =
 [i | i in Circuit where width[i] = width[r] /\ height[i] = height[r]]}
                    in if length(R) > 1 /\ min(R) = r then
                      forall(i in index_set(R) where i > 1)(
                  lex_less([y[R[i − 1]], x[R[i − 1]]], [y[R[i]], x[R[i]]]))
                                  else true endif)
```

4 Rotation Model

In this model,the only difference with the formal model is that the circuit's width and height and be exchanged.So an array of boolean variable is added which named Rotation.If the rotation[i] is true then it means the i^{th} circuit has been rotated and its width has been swapped with height.
There will be a new implied limitation in this model:if a circuit's height is larger than the plate's width,then it can not be rotated.

```
forall(i in circuit)(height[i] > w −> rotation[i] == false)
```

5 Search

According to the information in the minizinc tutorial,there are many choice for variable and constraint during search.In addition,it also contain restart strategy to solve the problem.

Variable choice:
- input_order
- first_fail
- Smallest
- dom_w_deg

Constraint choice:
- indomain_min
- indomain_median
- indomain_random
- indomain_split

Restart:
- restart_constant(100)

- restart_linear(100)
- restart_geometric(1.5,100)
- restart_luby(100)

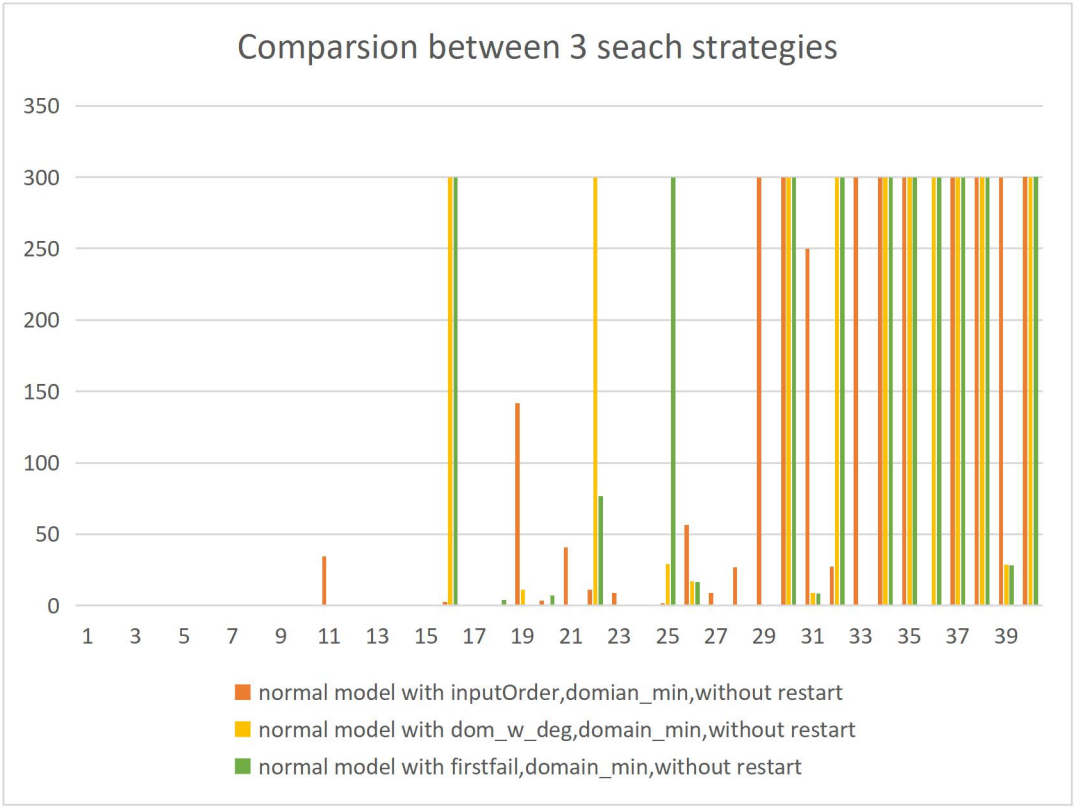
Some of them are combined to test during the search period.The search function has been built like the below one.

```
solve
  ::int_search([y[r] | r in circuitOrder], input_order, indomain_min,
complete)
  ::int_search([x[r] | r in circuitOrder], input_order, indomain_min,
complete)
  ::int_search([h], input_order, indomain_min, complete)
minimize h;
```

6 Result

6.1 Normal model

As the chart shown below,there are difference between input order,first_fail and dom_w_deg.And the input_order performs better than the other two.



6.2 Rotation model

As shown in the figure below,it is the result of the rotation model with input_order,indomain_min and without restart.Even though some instances takes less than 300 second to find the solution,it is not the best solution.This model is more complex than the normal one.

instances	timeElapse	solutions	failures	restarts
1	0.3	1	4	0
2	0.44	1	10	0
3	0.46	1	15	0
4	0.39	1	259	0
5	0.67	1	3441	0
6	0.43	1	1561	0
7	7.95	1	108913	0
8	1.02	2	6139	0
9	1.95	1	22407	0
10	1.02	3	3056133	0
11	0.34	2	3747051	0
12	5.73	3	3797454	0
13	0.29	2	4221347	0
14	2.81	3	2966360	0
15	0.54	22	4025161	0
16	1.04	2	2893606	0
17	0.51	2	3687982	0
18	300.04	0	2030541	0
19	0.3	1	2168220	0
20	0.34	1	2262891	0
21	0.43	2	3500355	0
22	2.55	2	1972887	0
23	0.66	36	4579660	0
24	300.04	0	1734917	0
25	0.74	1	1733621	0
26	300.08	0	1290623	0
27	0.31	2	2371166	0
28	0.38	2	2187736	0
29	0.37	2	2604640	0
30	277.67	2	1444461	0
31	300.04	0	1623852	0
32	300.04	0	1145364	0
33	23.14	47	3986330	0
34	76.24	26	4055382	0
35	0.27	1	1941350	0
36	0.29	11	4103327	0
37	300.04	0	1039952	0
38	300.05	0	1671487	0
39	300.04	0	1439499	0
40	300.09	0	237248	0

6.3 Comparison between normal model and rotation model

For comparison,the normal and rotation model both use search strategy input_order and domain_min with no restart.It looks that the normal model works better than the rotation model.