Practical Assignment: Part 1
2IMF25: Automated reasoning

M.R. Fortuin Student number: 0812105
m.r.fortuin@student.tue.nl
L.W.P.H. van Gansewinkel Student number: 0781263
l.w.p.h.v.gansewinkel@student.tue.nl

Contents

1 Assignment 1 **2**

2 Assignment 2 **4**

3 Assignment 3 **8**

4 Assignment 4 **10**

Chapter 1

Assignment 1

The first problem is solvable with yices. This can be expressed as an SMT problem with linear integer equations. In these linear equations N_i refers to the pallets of nuzzles on truck i, P_i refers to the pallets of prittles on truck i, S_i refers to the pallets of skipples on truck i, C_i refers to the pallets of crottles on truck i and D_i refers to the pallets of dupples on truck i.

The result of the assignment has to be subject too the following equations:

$$\sum_{i=0}^{6} N_i = 4 \qquad \text{4 nuzzles}$$

$$\sum_{i=0}^{6} S_i = 8 \qquad \text{8 skipples}$$

$$\sum_{i=0}^{6} C_i = 10 \qquad \text{10 crottles}$$

$$\sum_{i=0}^{6} D_i = 5 \qquad \text{5 dupples}$$

$$\forall_{1 \leq i \leq 6}\ 7800 \geq 700N_i + 800P_i + 1000S_i + 1500C_i + 100D_i \qquad \text{max weight on trucks}$$

$$\forall_{1 \leq i \leq 6}\ \neg(P_i > 0 \wedge C_i > 0) \qquad \text{no prittles and crottles together}$$

$$\forall_{1 \leq i \leq 6}\ D_i \leq 2 \qquad \text{maximum of 2 dupples}$$

$$\forall_{3 \leq i \leq 6}\ S_i = 0 \qquad \text{in truck 3 through 6 no skipples}$$

And maximization the following equation:

$$\sum_{i=0}^{6} P_i$$

Also implicit all variables has to be greater than zero, because it's impossible to load less then 0 crates.

Table 1.1: Pallets on the trucks

Truck	Nittles	Prittles	Skipples	Crottles	Dupples	Weight on Truck	Max Weight
1	0	7	2	0	1	7700	7800
2	0	2	6	0	2	7800	7800
3	2	8	0	0	0	7800	7800
4	0	0	0	5	0	7500	7800
5	0	0	0	5	2	7700	7800
6	2	8	0	0	0	7800	7800
Total	4	25	8	10	5		
Unit weight	700	800	1000	1500	100		

This is possible as input for yices, except for the maximize function. The maximal value of maximization function has to be found by trial and error. The maximum number of prittles that can be added is 25. The resulting load of the trucks can be found in table 1.1.

Chapter 2

Assignment 2

The second problem is solvable with yices. This can be expressed as an SMT problem with linear integer equations. This problem involves a couple of different variables, stemming from its two-dimensional nature. We require variables for the width and height of each component, including the chip itself. Next to this we, of course, need the positions of each component represented. We identify two variables SW and SH as the width and height of the chip, respectively. Similarly for the power components we have PW_i and PH_i as the width and height of the ith power component, respectively. We keep separate measurements for each power component to allow them to turn 90 degrees independently, while adding a minimal number of clauses to the problem. Finally we have CW_i and CH_i for the width and height of the ith regular component. As with the power components, these values for each component c_i can either be equal to the regular representation of c_i or the 90 turned representation. We also have the positions of the upper left corner for each power component and regular component respectively, given by PX_i, PY_i, CX_i and CY_i for the positions over the width and height of the chip, respectively. With the upper left corner, we mean that PX_i and PY_i are the lowest x and y coordinates on the chip that are taken up by power component p_i. We have one last variable in PD, indicating the distance the centres of two power components have to be from another in at least one dimension at $PD = 17$. For generating the problem in SMT, we use a variable n to denote the number of power components.

We now need to formulate each constraint on these variables. First and foremost, we must restrict the sizes of our components. Starting with the chip, we have the constraint:

$$SW = 29 \wedge SH = 22$$

Next, we have the power components, each of which has their own orientation, but the same restrictions to width and height options. Note that the form of this constraint allows you to create more options for differently

shaped power components by expanding with another or-clause:

$$\bigwedge_{i=0}^{n-1}(PW_i = 4 \wedge PH_i = 2) \vee (PW_i = 2 \wedge PH_i = 4)$$

Similarly, we get a constraint for each component with the optional 90 degree flip, we denote every component separately for now:

$$(CW_0 = 9 \wedge CH_0 = 7) \vee (CW_0 = 7 \wedge CH_0 = 9)$$
$$\wedge(CW_1 = 12 \wedge CH_1 = 6) \vee (CW_1 = 6 \wedge CH_1 = 12)$$
$$\wedge(CW_2 = 10 \wedge CH_2 = 7) \vee (CW_2 = 7 \wedge CH_2 = 10)$$
$$\wedge(CW_3 = 18 \wedge CH_3 = 5) \vee (CW_3 = 5 \wedge CH_3 = 18)$$
$$\wedge(CW_4 = 20 \wedge CH_4 = 4) \vee (CW_4 = 4 \wedge CH_4 = 20)$$
$$\wedge(CW_5 = 10 \wedge CH_5 = 6) \vee (CW_5 = 6 \wedge CH_5 = 10)$$
$$\wedge(CW_6 = 8 \wedge CH_6 = 6) \vee (CW_6 = 6 \wedge CH_6 = 8)$$
$$\wedge(CW_7 = 10 \wedge CH_7 = 8) \vee (CW_7 = 8 \wedge CH_7 = 10)$$

We now have the constraints for the sizes and 90 degrees turning.

We also require a constraint making sure that every component lies on the chip in its entirety, we do this the same way for both the power and regular components:

$$\bigwedge_{i=0}^{n-1}(PX_i \geq 0 \wedge PY_i \geq 0 \wedge PX_i + PW_i \leq SW \wedge PY_i + PH_i \leq SH)$$

$$\wedge \bigwedge_{i=0}^{7}(CX_i \geq 0 \wedge CY_i \geq 0 \wedge CX_i + CW_i \leq SW \wedge CY_i + CH_i \leq SH)$$

Next, we need a constraint for components, including power components not overlapping. This means that the origins of any components c_i and c_j must satisfy one of a few things. For one, c_j must lie entirely right of c_i, hence the x-coordinate of its origin must be at least cw_j away from c_i. Similarly for being left from c_i, it must be cw_i away. The same general trait applies to above and below with the y-coordinates and heights. Note that we check each component pair once and do not consider the reverse pair. This could also have been done by considering all pairs in each direction and only accounting for one vertical and one horizontal constraint each time. Finally, we also skip the pairs between power components, as they already require a more significant distance from one another. This gives us the following set of constraints:

$$\bigwedge_{i=0}^{7}\bigwedge_{j=i+1}^{7} CX_i \geq CX_j + CW_j \vee CX_i \leq CX_j - CW_i \vee CY_i \geq CY_j + CH_j \vee CY_i \leq CY_j - CH_i$$

$$\wedge \bigwedge_{i=0}^{7} \bigwedge_{j=0}^{n-1} CX_i \geq PX_j + PW_j \vee CX_i \leq PX_j - CW_i \vee CY_i \geq PY_j + PH_j \vee CY_i \leq PY_j - CH_i$$

To expand on that last statement, we will now consider the distances between power components. For each power component we must look at the center for checking this distance, so for a component c_i we take the coordinates $cx_i + cw_i/2$ and $cy_i + ch_i/2$. For two components c_i and c_j, these positions must be PD apart in either the x or y direction. Again, we only need to check once per pair, order of the pair does not matter:

$$\bigwedge_{i=0}^{n-1} \bigwedge_{j=i+1}^{n-1} (PX_i + PW_i/2 \geq PX_j + PW_j/2 + PD \vee PY_i + PH_i/2 \geq PY_j + PH_j/2 + PD$$

$$\vee PX_i + PW_i/2 \leq PX_j + PW_j/2 - PD \vee PY_i + PH_i/2 \leq PY_j + PH_j/2 - PD)$$

Finally, we get the last and most involved constraint, that each regular component must touch a power component. This is done by forcing the power component to be directly to either side of the component in some dimension and have at least 1 unit of overlap in the other dimension.

$$\bigwedge_{i=0}^{7} \bigvee_{j=0}^{n-1} ((PX_j = CX_i + CW_i \wedge PY_j > CY_i - PH_j \wedge PY_j < CY_i + CH_i)$$

$$\vee (PY_j = CY_i + CH_i \wedge PX_j > CX_i - PW_j \wedge PX_j < CX_i + CW_i)$$

$$\vee (PX_j = CX_i - PW_i \wedge PY_j > CY_i - PH_j \wedge PY_j < CY_i + CH_i)$$

$$\vee (PY_j = CY_i - PH_j \wedge PX_j > CX_i - PW_j \wedge PX_j < CX_i + CW_i))$$

For the full problem we simply conjunct all the previously shown constraints in this section. We will refrain from putting these together once more, as the problem would exceed half a page in size.

The program we used to generate the SMT problem, in addition to allowing the change of widths, heights and distances, also allows easy addition of more components.

After running the problem on yices and charting out the results, we got figure 2. The entire image shows a grid that is the entire chip. We marked all power components in grey and with a P. Each other component is given the number it has in the sequence it was introduced, as well as a distinct color.

The computation time, due to the relatively large set of clauses causes a noticeable delay in yices. This contrasts with the other 3 assignments, of which none had any visible delay. Applying yices these problems using higher values in the generalization variables might give us more insight into the scalability of these problems.

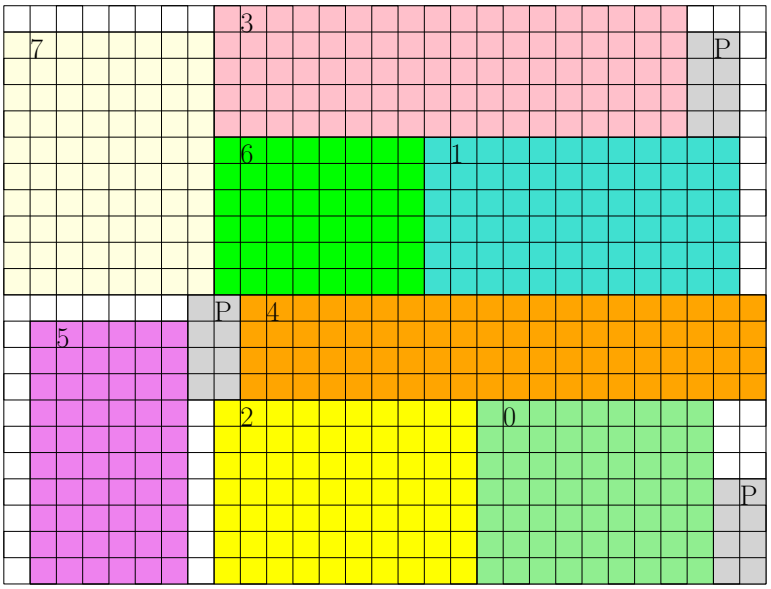

Figure 2.1: The chip with all required components.

Chapter 3

Assignment 3

The first problem is solvable with yices. This can be expressed as an SMT problem with linear integer equations. In these linear equations S_i refers to the start of job i with length i.

To express this problem in an SMT problem, 2 general predicate are defined. The first predicate makes sure that job a is after job b. This can be expressed as follows: $S_a \geq S_b + b$. This predicate is called $precedence(a, b)$. The second predicate is that 2 jobs, namely a and b, do not overlap. This is called $dont\_overlap(a, b)$ and can be expressed as follows:

$$\begin{aligned} &\neg(S_a \geq S_b \wedge S_a < S_b + b) \\ \wedge&\neg(S_a + a > S_b \wedge S_a + a \leq S_b + b) \\ \wedge&\neg(S_b \geq S_a \wedge S_b < S_a + a) \\ \wedge&\neg(S_b + b > S_a \wedge S_b + b \leq S_a + a) \end{aligned}$$

Then the resulting problem becomes quite easy. We assume there is a list P with all the precedences. Also a list C with all the jobs that can not run concurrently.

$$\begin{aligned} &\forall_{<a,b>\in P}\ precedence(a, b) \\ &\forall_{<a,b>\in C}\ dont\_overlap(a, b) \\ &\forall_{1\leq i\leq 12}\ S_i \geq 0 \end{aligned}$$

The total resulting span of the jobs is 36. The resulting plan is plotted in 3.1.

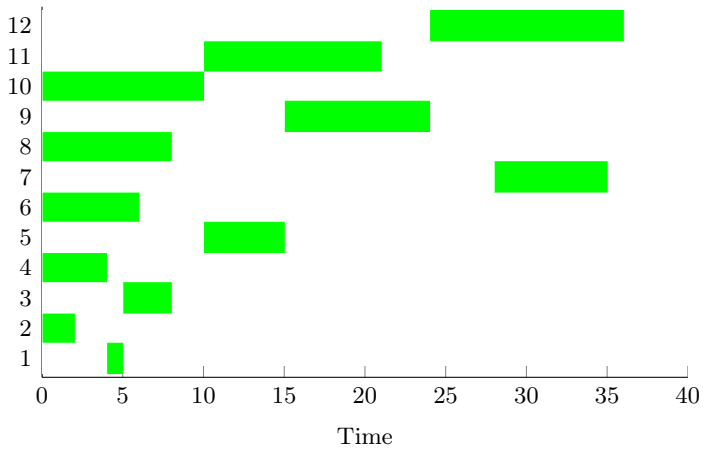

Figure 3.1: Planning results

Chapter 4

Assignment 4

The fourth and last problem is solvable with yices. This can be expressed as an SMT problem with linear integer equations. In these linear equations $A_{i,j}$ refers to the state of the integer variable a_i after j so-called 'steps'. Here the number i satisfies $1 \leq i \leq n$ where $n = 7$ as per description, and j starts at 0 and counts up as required. Since the rule for stepping states that $a_i = a_{i-1} + a_{i+1}$, we note that applying this rule for the same i in succession does not change a_i any further. This is a useful property, since the number of steps s to which j will count needs to be fixed somewhere, giving us $0 \leq j \leq s$. We can check whether it is possible to satisfy the constraints for the final state after exactly s steps, since there is the option to skip steps in this manner. It is also important that the step can only be applied to a variable a_i with $1 < i < n$, since the endpoints have only one neighbour.

Another feature to note about the step formula is that its result is only affected by a_{i-1} and a_{i+1}. We abuse this by allowing the program to perform multiple steps at a time, as long as no variable a_i changed in a step j affects the value of another variable a'_i in j. This information gives us the constraint that for every given step, a variable a_i either stays the same, or changes, forcing its neighbours a_{i-1} and a_{i+1} to stay the same:

$$\bigwedge_{i=2}^{n-1} A_{i,j} = A_{i-1,j-1} + A_{i+1,j-1} \vee A_{i,j} = A_{i,j-1}$$

Of course, this property must be applied to every j, $0 < j \leq s$, giving us the following constraint:

$$\bigwedge_{j=1}^{s} \bigwedge_{i=2}^{n-1} A_{i,j} = A_{i-1,j-1} + A_{i+1,j-1} \vee A_{i,j} = A_{i,j-1}$$

Aside from this, we must assure that a_1 and a_n do not take on random values to suit the problem, hence we must keep them from changing with

the following constraint:

$$\bigwedge_{j=1}^{s} A_{1,j} = A_{1,j-1} \wedge A_{n,j} = A_{n,j-1}$$

Finally, we need the variables to satisfy the constraint that by the end two must be equal to one another and $\geq m$, where $m = 50$ in our case, denoted by MIN in the program. Since a_1 and a_n do not change with any step, we can ignore them in this constraint. It is also worth noting that equality is symmetrical, so if we have checked a pair (a_i, a_k), we no longer bother with the pair (a_k, a_i). Additionally, if $a_i = a_k$ and $a_i \geq m$, it follows that $a_k \geq m$, so we can skip that constraint as well:

$$\bigvee_{i=2}^{n-1} \bigvee_{k=i+1}^{n-1} A_{i,s} = A_{k,s} \wedge A_{i,s} \geq MIN$$

$$\wedge MIN = 50$$

Throwing all these constraints together, we get the generalised complete constraint:

$$\bigwedge_{j=1}^{s} \bigwedge_{i=2}^{n-1} A_{i,j} = A_{i-1,j-1} + A_{i+1,j-1} \vee A_{i,j} = A_{i,j-1}$$

$$\wedge \bigwedge_{j=1}^{s} A_{1,j} = A_{1,j-1} \wedge A_{n,j} = A_{n,j-1}$$

$$\wedge \bigvee_{i=2}^{n-1} \bigvee_{k=i+1}^{n-1} A_{i,s} = A_{k,s} \wedge A_{i,s} \geq MIN$$

$$\wedge MIN = 50$$

This constraint can be used for any n and m (MIN). You are, however, required to adjust s to create more variables to make the problem satisfiable, which has to be done by generous overestimation or trail-and-error.

As resulting progression of values we have the table 4.1 calculated by our program. To show that this is a valid progression, we decomposed it into single steps as seen in 4.2, which would be the conventional solution.

Table 4.1: Progression of the variables using the described system

Phase	a_1	a_2	a_3	a_4	a_5	a_6	a_7
0	1	2	3	4	5	6	7
1	1	2	6	4	10	6	7
2	1	7	6	16	10	17	7
3	1	7	23	16	10	17	7
4	1	24	23	33	10	17	7
5	1	24	23	33	50	17	7
6	1	24	57	33	50	57	7

Table 4.2: Derived progression of the variables according to the rules

Phase	a_1	a_2	a_3	a_4	a_5	a_6	a_7
0	1	2	3	4	5	6	7
1	1	2	6	4	5	6	7
2	1	2	6	4	10	6	7
3	1	7	6	4	10	6	7
4	1	7	6	16	10	6	7
5	1	7	6	16	10	17	7
6	1	7	23	16	10	17	7
7	1	24	23	16	10	17	7
8	1	24	23	33	10	17	7
9	1	24	23	33	50	17	7
10	1	24	57	33	50	17	7
11	1	24	57	33	50	57	7