

# Practical Assignment

## Automated Reasoning 2IW15

prof dr H. Zantema  
Technische Universiteit Eindhoven  
HG room 6.73  
email: [h.zantema@tue.nl](mailto:h.zantema@tue.nl)  
August, 2011

### The programs to be used

Recommended programs to be used:

- **yices**: a program for satisfiability modulo theories (smt). It accepts smt format, but can also be used as a boolean SAT solver accepting dimacs format.
- **bddsolve**: a program based on BDDs. It can be used for satisfiability and counting the number satisfying assignments, and reachability.

For downloading these tools, description of the syntax and examples we refer to <http://www.win.tue.nl/~hzantema/ar.html>

Each of the problems can be solved using one of these two tools. Several similar tools are freely available; if you prefer other SMT solvers or other SAT solvers, or a tool for pseudo boolean constraints like `minisat+`, this is allowed too.

### The assignment

The practical assignment has to be executed by one or two persons. It consists of two parts.

The results of the assignment have to be described in two reports that should be handed in on paper. For the report on the first part the deadline is **November 14, 2011**, for the report on the second part the deadline is **January 9, 2012**.

For all used formulas an extensive documentation is required, explaining the approach and the overall structure. A generic approach is preferred, since this may result in clearer descriptions, increasing the confidence in the correctness of the results. Formulas of more than half a page should not be contained in the report, instead the structure of the formula should be explained. From the output of the programs relevant parts should be contained in the report. The answers on the problems should be motivated. Every report should contain name, student number and email address of each of the authors. In case of two authors each of them is considered to be responsible for the full text and all results.

Guidelines for grading:

- Clear and generic descriptions are appreciated, both of the formulas themselves and the way they were designed. An example of appreciated style is given at <http://www.win.tue.nl/~hzantema/prvb.pdf>.
- All results should be correct to obtain a 7.
- Reasons for obtaining higher than a 7 may be:
  - preference for harder problems,
  - remarkably clear and structured descriptions,
  - approaches allowing generalizations,
  - original approaches and solutions, or
  - more than the required number of problems solved.

## The problems for the first part

For the first part (deadline November 14, 2011) you have to describe solutions of

**at least 3 of the following 4 problems,**

using the indicated programs.

1. Six trucks have to deliver pallets to a partycentre. Every truck has a capacity of 8,500 kg and can carry at most eight pallets. In total, the following has to be delivered:
  - Four pallets of cheese, each having weight 700 kg.
  - At least fifteen pallets of beer, each having weight 1300 kg.
  - Eight pallets of wine, each having weight 1000 kg.
  - Ten pallets of softdrinks, each having weight 1500 kg.
  - Five pallets of potato chips, each having weight 100 kg.

Investigate what is the maximum number of pallets of beer that can be delivered, and show how for that number all pallets may be divided over the six trucks.

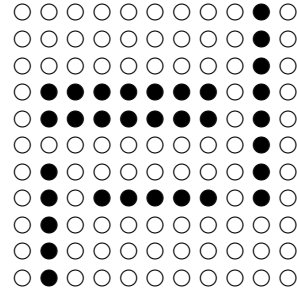
2. Square solitaire. In solitaire stones are put on a grid. A step in solitaire is as follows: if the position behind two consecutive stones is empty, then the first stone may jump over the second one, and the second one is removed. This may be done horizontally or vertically, but not in diagonal directions.

We consider a square grid of size  $5 \times 5$ , and consider the initial configuration where all positions are occupied except for the middle position.

- Is it possible to end with three stones?
  - Is it possible to end with two stones?
  - Is it possible to end with only one stone?
3. Eleven stones have to be put on the 121 positions  $(x, y)$  in the plane for which  $x, y \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$ , according the following rules:

- no two occupied positions have the same  $x$  value;
- no two occupied positions have the same  $y$  value;
- if  $(x, y)$  and  $(x', y')$  are distinct occupied positions, then  $(x - x')^2 + (y - y')^2 > 6$ ;
- the following positions (drawn black in the picture) are not occupied:

- $(2, y)$  for  $y \leq 5$ ;
- $(x, 4)$  for  $4 \leq x \leq 8$ ;
- $(10, y)$  for  $4 \leq y \leq 11$ ;
- $(x, y)$  for  $2 \leq x \leq 8$  and  $7 \leq y \leq 8$ .



In how many ways this can be done?

4. A rectangle of size  $18 \times 12$  has to be filled by squares: one square of size  $7 \times 7$ , one square of size  $6 \times 6$ , two squares of size  $5 \times 5$ , three squares of size  $4 \times 4$  and  $n$  squares of size  $3 \times 3$ . Establish what is the largest  $n$  for which this is possible.

## The problems for the second part

For the second part (deadline January 9, 2012) you have to describe solutions of the first problem and at least one of the remaining two problems, using the indicated programs.

1. This is an encoding of Chang and Roberts leader election protocol, see for instance <http://en.wikipedia.org/wiki/Chang-and-Roberts-algorithm>.

Among  $N$  nodes a leader has to be chosen. Every node has a unique identification (UID) represented by an integer, so distinct nodes have distinct UIDs. The nodes communicate by a ring structure: every node can pass a message MESS (being an integer) to its left neighbour, together with a bit MA stating whether the message is available, and a bit LA stating whether the leader is available. Further, every node has an internal bit PART stating whether it is participating, and an internal integer variable LEAD to store the UID of the chosen leader. Initially all bits MA, LA and PART are false. For every node  $i$  let  $n(i)$  be its left neighbour. Every node  $i$  may do steps according to the following rules.

- If both PART( $i$ ) and LA( $i$ ) are false, then  $i$  may start up by making PART( $i$ ) true and sending its UID, that is, MA( $n(i)$ ) is set to true and MESS( $n(i)$ ) is set to UID( $i$ ).
- If LA( $i$ ) is false and both PART( $i$ ) and MA( $i$ ) are true, and MESS( $i$ ) > UID( $i$ ), then  $i$  may forward its message, that is, MA( $n(i)$ ) is set to true and MESS( $n(i)$ ) is set to MESS( $i$ ), while in the same time MA( $i$ ) is set to false.
- If LA( $i$ ) is false and both PART( $i$ ) and MA( $i$ ) are true, and MESS( $i$ ) = UID( $i$ ), then  $i$  may decide to be the leader, that is, LA( $i$ ) is set to true, while MA( $i$ ) is set to false. The intuition is that this only will happen if this message originates

from  $i$  itself and has traversed the whole ring. Since for every node messages are only forwarded if they exceed the UID, this traverse may only occur if this UID is greater than all other UIDs.

- If  $LA(i)$  and  $PART(i)$  are true, then  $i$  may forward its knowledge about leadership and end its participation, that is,  $PART(i)$  is set to false, both  $LEAD(i)$  and  $MESS(n(i))$  are set to  $MESS(i)$ , and both  $LA(n(i))$  and  $MA(n(i))$  are set to true. Moreover,  $MA(i)$  is set to false.
- The algorithm successfully finishes as soon as there is a node  $i$  such that both  $LA(i)$  and  $MA(i)$  are true and  $PART(i)$  is false.

Fix  $N = 4$ .

- Prove that after finishing successfully, the value of  $LEAD(i)$  is the same for all nodes  $i$ . Show how many iterations are required in the corresponding analysis.
  - Prove that this resulting value of  $LEAD$  is equal to the maximum of all UIDs.
  - Prove that no deadlock may occur, that is, as long as the system has not finished successfully then always at least one rule is applicable.
2. Sorting. Twelve integer values  $a_1, \dots, a_{12}$  have to be sorted, i.e., they have to be pairwise swapped until  $a_i \leq a_j$  for all  $i < j$ . The basic routine to be used for this is  $\text{sort}(i, j)$ , that swaps  $a_i$  and  $a_j$  if  $a_j < a_i$ , and does not change the values otherwise. Determine the smallest value of  $k$  for which

```

for j := 1 to k do
  begin
    for i := 1 to 6 do sort(i,13-i);
    for i := 1 to 11 do sort(i,i+1);
  end

```

is a correct sorting routine. Also determine initial values for  $a_1, \dots, a_{12}$  for which the result is not sorted after execution of this routine for  $k$  being one less.

- Give a precise description of a non-trivial problem of your own choice, and encode this in such a way that it can be solved automatically by one of the given programs.