Constraint Satisfaction Problems

B. Nebel, C. Becker-Asano, S. Wölfl — University of Freiburg
Wintersemester 2014/15 — Department of Computer Science

Project: Part 2

Due: 19.12.2014
(4 + 12 + 4 points)

Task 1: Backtrack algorithm (4 points) Implement the algorithm Backtrack given in Figure 1 (or an iterative version of it).

```
Backtrack(N,a)

Input:  a constraint network N = <V,D,C>
        a partial solution a of N
        (possible: the empty instantiation a={})
Output: a solution of N or inconsistent

01.   IF a is defined for all variables in V THEN
02.      RETURN a
03.   ELSE select a variable vi for which a is not defined
04.      Di' := Di
05.      WHILE Di' is not empty
06.         select and delete a value x from Di'
07.         set vi to x
08.         a' := a + vi
09.         IF a' is consistent THEN
10.            a'' := Backtrack(N,a')
11.         ENDIF
12.         IF a'' is not inconsistent THEN
13.            RETURN a''
14.         ENDIF
15.      ENDWHILE
16.      RETURN inconsistent
17.   ENDIF
```

Figure 1: Backtrack Algorithm

Variables and values (line 03 and line 06, resp.) should be picked in lexicographic order (alternatives to this will be considered on the next sheet). Be careful with line 10 of the algorithm; you need to pass the constraint network to the next recursive invocation. On future project sheets we will have algorithms that refine the network during search, which may require to implement passing the network in a smarter way, e.g., by manually keeping track of the changes to the network.
Again we expect that all your Python code is contained in some top-level directory of your repository, except a single Python script *solver.py*. To run the script on an input file *some_path/input.xml* we will use the call *python3 solver.py some_path/input.xml* in the root directory of your repository. The solver should then call the implemented backtracking procedure on the input network from the XCSP file. After the search terminates your solver must print whether a problem instance is satisfiable (SAT) or unsatisfiable (UNSAT) and the time the solver used, e.g.:

```
SAT
Time: 213.46s
```

Task 2: Arc consistency (3×4 points) We now consider improvements to this solver by preprocessing networks once before the search. You may assume that all domains are sets of integers. Implement the following procedures in your solver:

- AC3
- AC2001
- GAC3

For AC3 and AC2001 domain filtering is only performed by considering binary constraints in the network. For GAC3 the filtering also considers higher-arity constraints. The algorithms are given in the lecture (see chapter 4 and the cited literature).
Implement command line options that allow for choosing the pre-processing algorithm as follows:

```
python3 solver.py --consistency=algo instance.xml
```

That is, the following command lines would pre-process the instance with the associated algorithm and then apply the backtrack search algorithm on the refined network:

```
python3 solver.py --consistency=none instance.xml
python3 solver.py --consistency=ac3 instance.xml
python3 solver.py --consistency=ac2001 instance.xml
python3 solver.py --consistency=gac3 instance.xml
```

The command without any consistency algorithm specified should still work:

```
python3 solver.py instance.xml
```

should perform no preprocessing.

Task 3: Comparing consistency algorithms (4 points) Add to your repository the file *project02_results.txt* This file must contain a table with the running times of the instances we provided according to the consistency algorithm implemented (AC3, AC2001, GAC3, and none) together with the information if the instance has a solution (satisfiable/unsatisfiable). Choose a reasonable time limit (e.g. 5 minutes) for your experiments, i.e., abort the solver if the runtime exceeds the time limit. If the solver cannot solve one of the instances mark this in the table (see below). The table must have the following format:

```
------------------------------------------------------------
instance01.xml   SAT
none      TIMEOUT
ac3        20.54s
ac2001     10.76s
gac3       15.54s

instance02.xml   UNSAT
none       50.32s
ac3         1.64s
ac2001      1.76s
gac3        2.74s


...
------------------------------------------------------------
```

Below the table, write a short comment in which you compare the results of your experiments.