Praktikum zur Vorlesung

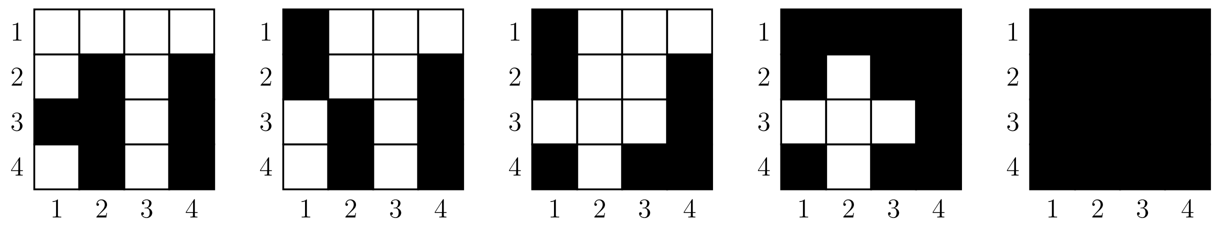
Computational Intelligence

Prof. Torsten Schaub, Laferrière François, Cedric-John Martens, Jan Heuer

Universität Potsdam (Wissensverarbeitung und Informationssysteme) — Sommersemester 2020

Praktikumsaufgabe 2 (Blackout)

Problem Description. The task of this project is to solve a grid puzzle, whose goal is to color all grid cells black, using ASP. To illustrate this, consider a sequence as follows:



Given the initial grid with black and white cells on the left, the grid cells can be manipulated in turns by picking (exactly) one grid cell at a time. The picked as well as its horizontally and vertically adjacent cells then switch their colors, from white to black, or vice versa. For instance, the cell at position $(2, 1)$ is picked first, so that itself and the neighbors at $(1, 1)$, $(2, 2)$, $(3, 1)$ switch their colors, leading to the second grid in the sequence. By in the sequel picking the cells at $(4, 2)$, $(1, 3)$, and $(3, 2)$, one after the other, all grid cells become black and the puzzle is solved within four turns.[1]

Representation in ASP. An instance like the above is specified by facts as follows:

```
cell(1,1).  cell(1,2). ...  cell(3,4).  cell(4,4).
black(2,2).  black(2,4).  black(3,1).
black(3,2).  black(3,4).  black(4,2).  black(4,4).
time(1).  time(2).  time(3).  time(4).
```

Facts over the predicate `cell/2` provide the coordinates of grid cells. Those over `black/2` determine the cells colored black in the initial grid, while any remaining cell is initially white. Moreover, facts over `time/1` provide a sequence of consecutive integers starting from `1`, expressing the number of turns in which all grid cells shall be colored black.

The sequence of turns taken in a solution shall be provided by atoms over the predicate `switch/3` in a stable model, indicating the coordinates of one cell picked per turn. Along with the instruction '`#show switch/3.`', a correct implementation in file `blackout.lp` may yield the following `clingo` behavior, where `grid01.lp` includes the above facts and the (first) stable model represents the sequence shown above:

```
$ clingo-5.3.0 grid01.lp blackout.lp
clingo version 5.3.0
Reading from grid01.lp ...
Solving...
Answer: 1
switch(2,1,1) switch(4,2,2) switch(1,3,3) switch(3,2,4)
SATISFIABLE
```

[1] To further familiarize with the puzzle, you may try it out at http://www.logicgamesonline.com/lightsout/ or http://www.lojol.de/html/licht.html.

Framework. In the `blackout.zip` archive at Moodle you will find twelve example instances. You have to submit a file named `blackout.lp`, included as template in `blackout.zip`, that contains the following line (and no more `#show` statements), so that only atoms over the predicate `switch/3` appear in the output:

```
#show switch/3.
```

Formalities. You can work on the solution alone or in groups of three students. Different groups have to submit different solutions, in case of plagiarism all groups involved will fail the project. Please submit your encoding by **Friday, Jully 31, 2020** via YETI. (All group members have to create a YETI account!) Be sure to submit your encoding in a file named `blackout.lp`, containing only lowercase letters in its name.

We will test your encoding with the twelve provided instances as well as twelve additional instances. Your program has to correctly encode *some* solution for every instance. (In fact, our test instances usually have plenty solutions, and all instances we use are satisfiable.) This will be tested automatically by YETI after you uploaded the encoding (with a slight delay). If your encoding is incorrect, YETI will display an error message (output '`failure`'). We expect that at least 16 out of the 24 test instances at YETI are solved within the time and memory limits (YETI output '`success`'), and none must be answered incorrectly. Please correct any errors that occur on your own or contact us if you get stuck.

Tips:

- Note that the four directions (up, down, left, right) are entirely symmetric and vary only by coordinate differences between cells. Please avoid code duplication due to referring to directions more specifically than needed; e.g., distinguishing directions by using different names for auxiliary predicates is not a good idea.
- A command-line call to inspect the ground instantiation looks like the following:

  ```
  $ clingo-5.4.0 grid01.lp blackout.lp --text
  ```

- Problem instances get increasingly harder, the larger the given grid and/or the number of turns is. You may presumably improve the solving performance by encoding a canonical order in which cells are picked, since the final color (i.e., black) of a cell is only determined by the number of turns in which its color is switched, while the order of such turns does not make any difference.
- If you are stuck, please contact us. We will do our best to answer all your questions. You can send us questions and remarks via Moodle (forum), or send them via mail to `ci@lists.cs.uni-potsdam.de`.
- Start as soon as possible to avoid running out of time. However, if you still realize that you have problems making it before the deadline, please contact us instead of copying another solution.