Logical reasoning and programming, lab session 9

(November 23, 2020)

For the following experiments use `clingo`. An online version of clingo is sufficient. Moreover, the examples mentioned bellow are available there. However, for further experiments, it is recommended to install clingo. A convenient way is by using `conda install -c potassco clingo` from Anaconda or Miniconda. Anyway, be sure that you have at least version 4 which uses the ASP-Core-2 format.

9.1 Find all the minimal models of

(a) $\{p \leftarrow q. \quad q \leftarrow p.\}$,

(b) $\{p \mid q. \quad r \leftarrow p.\}$,

(c) $\{p \mid q. \quad r \leftarrow p. \quad s \leftarrow q.\}$.

9.2 If two (positive) logic programs are equivalent, meaning they have the same models, then they have the same minimal models. Does the opposite implication hold? Prove, or provide a counter-example.

9.3 Find all the stable models of

(a) $\{p \leftarrow \mathrm{not}\, q. \quad q \leftarrow \mathrm{not}\, p.\}$,

(b) $\{p. \quad q. \quad r \leftarrow p, \mathrm{not}\, s.\}$,

(c) $\{p \mid q. \quad r \leftarrow \mathrm{not}\, p.\}$,

(d) $\{p \leftarrow \mathrm{not}\, q. \quad q \leftarrow \mathrm{not}\, p. \quad p \leftarrow q. \quad q \leftarrow p.\}$.

In (d) find also all the minimal models.

9.4 Check the Harry and Sally example.

9.5 Check the Flying Birds example and pay special attention to the use of negations. Try to add `bird(joe)`. Does something change if we extend our knowledge by adding `penguin(joe)`?

9.6 Find all the stable models of $\{p \leftarrow \mathrm{not}\,\mathrm{not}\, q. \quad q \leftarrow \mathrm{not}\,\mathrm{not}\, p.\}$.

9.7 Check the Traveling Salesperson example. Note that the minimize line is equivalent to

```
:~ cycle(X,Y), cost(X,Y,C). [C,X,Y]
```

discussed during the lecture. Does it make any difference if we change `[C,X,Y]` to `[C]`?

9.8 Write a general solver for graph coloring and then check it against the Graph Coloring example. Assume that `n` contains the number of available colors and the input is given by predicates `node/1` and `edge/2` describing the names of nodes and edges between them, respectively. Compare this solution with your SAT solution.

9.9 Guess how many lines of code you need to solve the n-queens problem and then briefly check the solution.

Hint: The description of diagonal constraints is a well-known trick described for example here.

9.10 Check the Blocksworld Planning example. You can find how the incremental solving works and a brief description of the solution in Potassco guide and further inputs in examples.