Computational Logic and Automated Reasoning
Winter 2017 Take Home Examination

Name: **Student Number:**

Instructions: Answer succinctly and clearly. Explain your answers/solutions. Correct answers without a valid justification count as wrong. Submit by email (to the instructor) a single document in PDF format prepared with Latex. Include the problem statements with your solutions. Departing from the instructions will make you lose points.
By default every part of a problem has three points. You get 3 points if the solution is perfect, 2 if it is almost perfect, and 0, otherwise. No points in between.
Deadline: April 25 at 23:30.

1. (a) Assume that you have a finite set of ground facts of the form $R(a, b)$, with arbitrary constants. Assume that they all become the facts for a Prolog program. You want to check via Prolog if the relation R is symmetric (you do not do anything yourself about checking the property). How would you proceed? Explain if you would add something to the program (and what) and what queries would you pose. Justify. [3 points]
(b) The same as in (b), but about transitivity. [3 points]

2. Consider a relational database with the following tables (or set of facts if you want):

R	A	B
	a	b
	a	c
	c	b
	d	e

S	B	C
	a	b
	a	c
	c	b
	d	a

This database does not satisfy the sentence $\forall x \forall y(R(x, y) \rightarrow \exists z S(y, z))$. It requires that every value appearing in the second column in R has to appear in the first column of S accompanied by some value. (In databases this is called a "referential integrity constraint".)

(a) Define by means of a Prolog program a predicate $V(\cdot, \cdot)$ that collects all the pairs (x, y) from R for which y does not appear in S as indicated above. [3 points]

(b) Show how you would use Prolog and predicate V to verify if the integrity constraint is violated. Show the execution of Prolog as you understand it, and what you get from it (this is not about running Prolog). [3 points]

(c) You pose to the database above the query in FO predicate logic: [3 points]

$$\mathcal{Q}(x): \ \exists y \exists z(R(x, y) \wedge S(y, z) \wedge \neg S(z, y)).$$

Evaluate the query, i.e. find the data values v for variable x, such that: $D \models \mathcal{Q}[v]$, **by strictly and compositionally applying the inductive definition of formula satisfaction of FO predicate logic** (no other method is valid).

3. Give in detail an explicit model (i.e. a satisfying structure) for the set of sentences Σ below where the extension (contents) of *Above* **is not** the transitive closure of the extension of *On*.

$\Sigma = \{\forall x \forall y(\exists z(Above(x,z) \wedge On(z,y)) \rightarrow Above(x,y)),\ \forall x \forall y(On(x,y) \rightarrow Above(x,y))\}$.
(a) Verify that the given structure does satisfy Σ, and **(b)** Verify the non-transitive closure as explained above. [3 points each part].

4. **(a)** Define in first-order predicate logic the symmetric closure of a given binary relation $R(\cdot,\cdot)$. For this introduce a new predicate S. [3 points]

(b) Do the same as in (a), but using Prolog. [3 points]

5. You want to use an answer set program (ASP) for 3-Graph-Coloring (3-GC), that you saw in class, to solve a problem about propositional SAT (with formulas in CNF). So, you have to *reduce* SAT to 3-GC. (This is how you prove that, given that SAT is NP-hard, that 3-GC is also NP-hard. Actually, it is good enough to use the NP-hardness of 3-SAT, about satisfiability of propositional formulas in CNF with each clause having at most 3 literals.) The appendix shows how to do the (general) reduction. It takes an instance for 3-SAT, i.e. a formula φ, and constructs an instance $G(\varphi)$ for 3-CG, i.e. a graph. It holds: φ is satisfiable iff G is colorable with 3 colors.

(a) Given the formula φ : $(p \vee \neg r \vee q) \wedge (\neg p \vee r \vee s) \wedge (\neg s \vee p \vee \neg q)$, construct the corresponding graph $G(\varphi)$. Explain the correspondence, and draw a picture of the graph. [3 points]

(b) Write an ASP to solve 3-GC for the resulting graph in (a). [2 points]

Run it with DLV and (clearly explaining) obtain from it an answer about the colorability of $G(\varphi)$ and the satisfiability of φ. Attach as an appendix the run with DLV (which implies that the main aspects of the solution should be shown in the main body of the submission). [5 points]

(c) Write an ASP that solves the instance of SAT in (a), directly, without the detour through 3-GC. The methodology should be general, not a hack for this particular instance. So, explain in what sense it is general. [3 points].

(d) Run the program in (c) on DLV, and give the solution to the decision problem accordingly. Also, obtain the answer sets (models) of the program and explain how you interpret them. Attach the run as an appendix (which implies that the main aspects of the solution should be shown in the main body of the submission). [5 points]

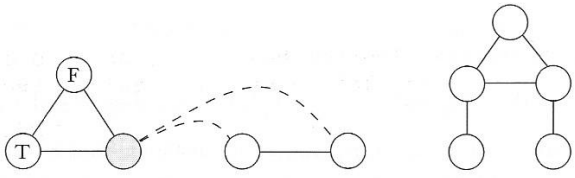
Appendix: Reducing 3-SAT to 3-GC

3COLOR is in NP because a coloring can be verified in polynomial time. We show *3SAT* $\leq_P$ *3COLOR*. Let $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_l$ be a 3cnf formula over variables $x_1, x_2, \cdots, x_m$, where the c_i's are the clauses. We build a graph G_ϕ containing $2m + 6l + 3$ nodes: 2 nodes for each variable; 6 nodes for each clause; and 3 extra nodes. We describe G_ϕ in terms of the subgraph gadgets given in the Hint.

G_ϕ contains a variable gadget for each variable x_i, two OR-gadgets for each clause, and one palette gadget. The four bottom nodes of the OR-gadgets will be merged with other nodes in the graph, as follows. Label the nodes of the palette gadget T, F, and R. Label the nodes in each variable gadget $+$ and $-$ and connect each to the R node in the palette gadget. In each clause, connect the top of one of the OR-gadgets to the F node in the palette. Merge the bottom node of that OR-gadget with the top node of the other OR-gadget. Merge the three remaining bottom nodes of the two OR-gadgets with corresponding nodes in the variable gadgets so that if a clause contains the literal x_i, one of its bottom nodes is merged with the $+$ node of x_i whereas if the clause contains the literal $\overline{x_i}$, one of its bottom nodes is merged with the $-$ node of x_i.

[illegible] that if ϕ is satisfi-

Show that *3COLOR* is NP-complete. (Hint: Use the following three subgraphs.)



palette variable OR-gadget