Logic and Reasoning in Computer Science SS 2025 **Exercise Sheet 1**

Instructions for project submissions: You are requested to solve at least one project problem and submit your project solutions by **April 11, 2025**. Project problems are explicitly stated as *Project Problem 1.1* and *Project Problem 1.2*. If you solve both project problems, the deadline for submitting your project solutions is still April 11, 2025.

Please upload only your project problem solutions (any solution to other problems will be ignored). For the project submission, please upload your solutions in a zip archive named using your student ID, e.g. `e123456789.zip`, using the following naming and formatting:

- `proj1.cnf` : Your encoding of Project Problem 1.1 in CNF if you solved it
- `proj1.out` : The dump of the SAT solver (MiniSAT, CaDiCaL,...) output when run on `proj1.cnf`
- `proj1.*` : A short interpretation/analysis of your SAT solver's output for Project Problem 1.1, concluding satisfiability or unsatisfiability of your project.
- `exam_scheduler.py` : Your solution of Project Problem 1.2 if you solved it.
- `proj2.*` : A short document explaining your solution. Detailed requirements are explained below.

A star $\star$ in the above file names is used to denote either a `pdf` or a `txt`, format. Please **follow the file naming convention** and submit your project solutions as instructed, by April 11, 2025. If the file names are incorrect, your projects may not be credited.

Exercise Problem 1.1. Are the following statements true? (Provide either a proof or a countermodel)

1. if $A \models B$ then $\neg A \models \neg B$
2. if $A \models B$ and $A \wedge B \models C$ then $A \models C$
3. if $A \vee B \models A \wedge B$ then $A \equiv B$.

Solution:
(note that other ways of solving the problem might be found!)

1. False. Countermodel: $A := \bot$, and $B := \top$
2. True. By contradiction. $A \models C$ is false only if there is an interpretation I s.t. $I(A) = 1$ and $I(C) = 0$, but then for $A \wedge B \models C$ to be true $I(B) = 0$, but then $A \models B$ cannot hold (as $I(A) = 1$ and $I(B) = 0$)
3. True. Assume by contradiction that $A \equiv B$ does not hold, this means that there is an interpretation I such that $I(A) = 0$ and $I(B) = 1$ (or viceversa). But then $A \vee B \models A \wedge B$ does not hold (as $I(A \vee B) = 1$ and $I(A \wedge B) = 0$).

Exercise Problem 1.2. Prove or disprove the correctness of the following reasoning using tableaux:

1. I can bake the cake only if I buy the ingredients today.

2. I can go to the gym only if I finish my homework early.

3. I cannot both buy the ingredients and finish homework early.

Therefore, either I will not bake the cake or I will not go to the gym.
Solution:
$C := bakethecake$, $I := buytheingredients$, $G := gotothegym$, and $H := finishhomeworkearly$.

1. $C \to I$

2. $G \to H$

3. $\neg(I \wedge H)$

The sentence $\neg C \vee \neg G$ is entailed by 1-3 (easy tableaux proof, or semantically, by postulating $I(\neg C \vee \neg G) = 0$, the interpretation I does not satisfy 1-3.

Exercise Problem 1.3. Three boxes are presented to you. One contains gold, the other two are empty. Each box has imprinted on it a clue as to its contents; the clues are:

Box 1 *The gold is not here*

Box 2 *The gold is not here*

Box 3 *The gold is in Box 2*

Only one message is true; the other two are false. Which box has the gold?

Solution:
Let B_i with $i \in \{1, 2, 3\}$ stand for "gold is in the i-th box". We can formalize the statements of the problem as follows:

1. One box contains gold, the other two are empty

$$(B_1 \wedge \neg B_2 \wedge \neg B_3) \vee (\neg B_1 \wedge B_2 \wedge \neg B_3) \vee (\neg B_1 \wedge \neg B_2 \wedge B_3)$$

2. Only one message is true; the other two are false.

$$(\neg B_1 \wedge \neg\neg B_2 \wedge \neg B_2) \vee (\neg\neg B_1 \wedge \neg B_2 \wedge \neg B_2) \vee (\neg\neg B_1 \wedge \neg\neg B_2 \wedge B_2)$$

Simplifying the formulas and making a truth table we can see that the only assignment that verifies both is $I(B_1) = 1, I(B_2) = I(B_3) = 0$, which implies that the gold is in the first box.

Exercise Problem 1.4. Let A be a propositional formula with $n \geq 1$ propositional variables such that:

- A is not a propositional atom, and
- A is built from propositional atoms using only $\neg$ and $\leftrightarrow$.

How many branches does a splitting tree of A have? Provide a sufficiently detailed explanation of your answer.

Solution:
The splitting tree of A has 2^n branches, where n is the number of propositional variables of A.

The proof is based on induction over the number of variables of A, by noting that a split over a variable in A reduces the number of variables of A to be split upon further only exactly by one.

Exercise Problem 1.5. Consider the formula:

$$(p \to (q \to r)) \wedge \neg((\neg q \vee r) \wedge \neg p).$$

Find a formula that is equivalent to the above formula and is shortest in size. Justify your answer.

Solution:
The formula $p \leftrightarrow (q \to r)$ is equivalent to the given formula. It is the shortest formula in size: it contains all three variables p, q, r and uses exactly two logical connectives. Note that any formula with three propositional variables has to use at least two logical connectives.

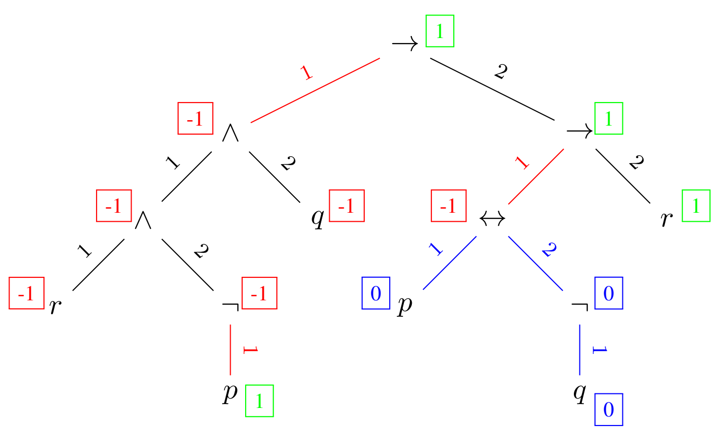
Exercise Problem 1.6. Draw the parse tree of the following formula:

$$r \wedge \neg p \wedge q \to ((p \leftrightarrow \neg q) \to r)$$

List all subformulas with negative polarity in the formula above.

Solution:

A parse tree of $r \wedge \neg p \wedge q \to ((p \leftrightarrow \neg q) \to r)$, after applying the coloring algorithm to compute polarities, is visualized below, with polarities of subformulas indicated in framed boxes:



The subformulas with negative polarity are:

- $r \wedge \neg p \wedge q$
- $r \wedge \neg p$
- r
- $\neg p$
- q
- $p \leftrightarrow \neg q$

Exercise Problem 1.7. Consider the formula:

$$(\neg p \to \neg r) \wedge ((r \wedge q) \to (p \leftrightarrow q))$$

(a) Which atoms are pure in the above formula?

(b) Compute a clausal normal form C of the above formula by applying the CNF transformation algorithm with naming and optimization based on polarities of subformulas.

(c) Decide the satisfiability of the computed CNF formula C by applying the DPLL method to C. If C is satisfiable, give an interpretation which satisfies it.

Solution.
(a) p and q cannot be pure due to the equivalence; r is pure and negative.
(b)

Name	Subformula	Definition	CNF
			n_1
n_1	$(\neg p \to \neg r) \wedge ((r \wedge q) \to (p \leftrightarrow q))$	$n_1 \to (n_2 \wedge n_3)$	$\neg n_1 \vee n_2$ $\neg n_1 \vee n_3$
n_2	$(\neg p \to \neg r)$	$n_2 \to (n_4 \to n_5)$	$\neg n_2 \vee \neg n_4 \vee n_5$
n_3	$((r \wedge q) \to (p \leftrightarrow q))$	$n_3 \to (n_6 \to n_7)$	$\neg n_3 \vee \neg n_6 \vee n_7$
n_4	$\neg p$	$\neg p \to n_4$	$p \vee n_4$
n_5	$\neg r$	$n_5 \to \neg r$	$\neg n_5 \vee \neg r$
n_6	$(r \wedge q)$	$(r \wedge q) \to n_6$	$\neg r \vee \neg q \vee n_6$
n_7	$(p \leftrightarrow q)$	$n_7 \to (p \leftrightarrow q)$	$\neg n_7 \vee \neg p \vee q$ $\neg n_7 \vee p \vee \neg q$

The final CNF is the following:

$$C = \left\{ \begin{matrix} n_1, & \neg n_1 \vee n_2, & \neg n_1 \vee n_3, & \neg n_2 \vee \neg n_4 \vee n_5, & \neg n_3 \vee \neg n_6 \vee n_7, \\ p \vee n_4, & \neg n_5 \vee \neg r, & \neg r \vee \neg q \vee n_6, & \neg n_7 \vee \neg p \vee q, & \neg n_7 \vee p \vee \neg q \end{matrix} \right\}$$

(c) We apply the DPLL procedure while building a satisfying assignment I:

1. Unit propagate n_1 and set pure literal $\neg r$, $I = \{n_1 \mapsto 1, r \mapsto 0\}$

$$\{n_2, \quad n_3, \quad \neg n_2 \vee \neg n_4 \vee n_5, \quad \neg n_3 \vee \neg n_6 \vee n_7, \quad p \vee n_4, \quad \neg n_7 \vee \neg p \vee q, \quad \neg n_7 \vee p \vee \neg q\}$$

2. Unit propagate n_2, $I = \{n_1 \mapsto 1, r \mapsto 0, n_2 \mapsto 1\}$

$$\{n_3, \quad \neg n_4 \vee n_5, \quad \neg n_3 \vee \neg n_6 \vee n_7, \quad p \vee n_4, \quad \neg n_7 \vee \neg p \vee q, \quad \neg n_7 \vee p \vee \neg q\}$$

3. Unit propagate n_3, $I = \{n_1 \mapsto 1, r \mapsto 0, n_2 \mapsto 1, n_3 \mapsto 1\}$

$$\{\neg n_4 \vee n_5, \quad \neg n_6 \vee n_7, \quad p \vee n_4, \quad \neg n_7 \vee \neg p \vee q, \quad \neg n_7 \vee p \vee \neg q\}$$

4. Assume $\neg n_7$ and unit propagate (i.e. branch into n_7 being false), $I = \{n_1 \mapsto 1, r \mapsto 0, n_2 \mapsto 1, n_3 \mapsto 1, n_7 \mapsto 0\}$

$$\{\neg n_4 \vee n_5, \quad \neg n_6, \quad p \vee n_4\}$$

5. Unit propagate $\neg n_6$, $I = \{n_1 \mapsto 1, r \mapsto 0, n_2 \mapsto 1, n_3 \mapsto 1, n_7 \mapsto 0, n_6 \mapsto 0\}$

$$\{\neg n_4 \vee n_5, \quad p \vee n_4\}$$

6. Assume $\neg n_4$ and unit propagate (i.e. branch into n_4 being false), $I = \{n_1 \mapsto 1, r \mapsto 0, n_2 \mapsto 1, n_3 \mapsto 1, n_7 \mapsto 0, n_6 \mapsto 0, n_4 \mapsto 0\}$

$$\{p\}$$

7. Unit propagate p, $I = \{n_1 \mapsto 1, r \mapsto 0, n_2 \mapsto 1, n_3 \mapsto 1, n_7 \mapsto 0, n_6 \mapsto 0, n_4 \mapsto 0, p \mapsto 1\}$

A possible satisfying assignment to the initial formula: $\{p \mapsto 1, q \mapsto 1, r \mapsto 0\}$

We can verify the assignment, by using for example the syntactic version of formula evaluations in an interpretation:

$$\begin{aligned}
(\neg\top \rightarrow \neg\bot) \wedge ((\bot \wedge \top) \rightarrow (\top \leftrightarrow \top)) \\
(\bot \rightarrow \top) \wedge (\bot \rightarrow \top) \\
\top \wedge \top \\
\top
\end{aligned}$$

Exercise Problem 1.8. Find a model of the formula $((\neg p \rightarrow q) \rightarrow p) \rightarrow \neg p$ using only the pure atom rule.

Solution:

One can note that the only occurrence of q is positive, therefore q can be replaced by $\top$ so that we obtain an equi-satisfiable formula

$$((\neg p \implies \top) \implies p) \implies \neg p.$$

This formula can be simplified to

$$p \implies \neg p.$$

Now both occurrences of p are negative, hence p can be replaced by $\bot$ obtaining

$$\bot \implies \neg\bot.$$

which can be simplified to $\top$.

Therefore, the original formula is satisfiable and $\{p \mapsto 0, q \mapsto 1\}$ is a model of this formula.

Exercise Problem 1.9. Apply the standard CNF transformation algorithm and the definitional transformation algorithm (both the non-optimized and optimized versions) to the following formula:

$$\neg((p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)).$$

Solution:

We give here only the definitional transformation algorithm for clausifying the given formula. We start with the *non-optimized definitional CNF transformation*, summarized below:

	subformula	definition	clauses (non-optimized)
			n_1
n_1	$\neg((p \to q) \leftrightarrow (\neg q \to \neg p))$	$n_1 \leftrightarrow \neg n_2$	$n_1 \vee n_2$
			$\neg n_1 \vee \neg n_2$
n_2	$(p \to q) \leftrightarrow (\neg q \to \neg p)$	$n_2 \leftrightarrow (n_3 \leftrightarrow n_4)$	$\neg n_2 \vee \neg n_3 \vee n_4$
			$\neg n_2 \vee \neg n_4 \vee n_3$
			$n_3 \vee n_4 \vee n_2$
			$\neg n_3 \vee \neg n_4 \vee n_2$
n_3	$p \to q$	$n_3 \leftrightarrow (p \to q)$	$\neg n_3 \vee \neg p \vee q$
			$p \vee n_3$
			$\neg q \vee n_3$
n_4	$\neg q \to \neg p$	$n_4 \leftrightarrow (\neg q \to \neg p)$	$\neg n_4 \vee q \vee \neg p$
			$\neg q \vee n_4$
			$p \vee n_4$

Observe that in the table above, n_1 has only positive polarity and n_2 has only negative polarity. Hence, instead of $n_1 \leftrightarrow n_2$ it suffices to have only $n_1 \to n_2$. Similary, instead of $n_2 \leftrightarrow (n_3 \leftrightarrow n_4)$ we only need $(n_3 \leftrightarrow n_4) \to n_2$ in the definitional CNF transformation. The subformula n_3 and n_4 have polarity 0. As a result, the *optimized definitional CNF transformation* is summarized below:

	subformula	definition	clauses (optimized)
			n_1
n_1	$\neg((p \to q) \leftrightarrow (\neg q \to \neg p))$	$n_1 \to \neg n_2$	$\neg n_1 \vee \neg n_2$
n_2	$(p \to q) \leftrightarrow (\neg q \to \neg p)$	$(n_3 \leftrightarrow n_4) \to n_2$	$n_3 \vee n_4 \vee n_2$
			$\neg n_3 \vee \neg n_4 \vee n_2$
n_3	$p \to q$	$n_3 \leftrightarrow (p \to q)$	$\neg n_3 \vee \neg p \vee q$
			$p \vee n_3$
			$\neg q \vee n_3$
n_4	$\neg q \to \neg p$	$n_4 \leftrightarrow (\neg q \to \neg p)$	$\neg n_4 \vee q \vee \neg p$
			$\neg q \vee n_4$
			$p \vee n_4$

Project Problem 1.1. Consider the following instance of the so-called pigeonhole problem: There are 3 pigeons and 2 holes. Is it possible to assign pigeons to holes so that every pigeon occupies some hole and no hole contains more than one pigeon?

(a) Formalize this instance of the pigeonhole problem as a propositional satisfiability problem.

(b) Encode the problem as an input to the MiniSat (or other SAT solver) solver in the DIMACS format[1] and evaluate the SAT solver on your encoding. Interpret the result of the SAT solver. Is the problem satisfiable? Provide the electronic version of your SAT solver encoding together with your solution.

Project Problem 1.2. After spending all spring break chasing Easter eggs, the faculty professors no longer have time to schedule exams. You and your peers fear that this will result in a disastrous result.
Students are never happy studying for exams. However, they are even less happy when they have to take two exams on the same day. Willing to show your mastery of propositional logic, you decide to compute a schedule to submit to the professors that will make everyone happy.
You are given a list of students and the exams they have to take. Your task is to find a satisfactory schedule using a SAT solver. That is, a schedule such that every exam is planned in a room with enough capacity for all students taking the exam while avoiding multiple exams on the same day. In other words:

- Each exam is planned exactly once, in a room with enough capacity for all students taking the exam.
- A room can host at most one exam at a time.
- No student has two exams on the same day.

Resources. You can download the project archive on TUWEL. This archive contains:

- A Jupyter notebook `exam_scheduler.ipynb` containing precise instructions on how to solve the task.
- A python file `exam_scheduler.py` containing the template for your project. Your task is to complete the functions `encode_cnf` and `decode_cnf` and submit this file only.
- A python file `scheduler_utils.py` containing useful functions to check the correctness of your encoding, as well as a pretty printer. There is no need to edit nor submit this file.
- A folder `2_rooms_3_days_5_courses_10_students` containing an example of scheduling problem. You are invited to create more to test your solution more thoroughly.

Libraries used. This project requires coding in Python. While it is possible that you have never programmed in this language before, it is a very useful language that is definitely worth learning. For this project, we do not evaluate your knowledge of the language.
The files make use of three different libraries.

- **Jupyter notebooks**[2]. Jupyter notebooks is a very useful tool to display intermediate results of a python script. VS Code[3] has very good support for notebooks, but you can also run them on the browser.

[1] `https://jix.github.io/varisat/manual/0.2.0/formats/dimacs.html`
[2] `https://jupyter.org/`
[3] `https://code.visualstudio.com/docs/datascience/jupyter-notebooks`

- **Pandas**[4]. Pandas is a library that allows to manipulate datasets conveniently. It is one of the most useful libraries for data science and representations tasks in Python. For this project, it is solely used as a convenient way to store and share data between functions. We only make use of the DataFrame. A DataFrame is an indexed table with named columns. The online documentation is very thorough. Any questions related to the manipulations of DataFrames can be searched online. If there is still some confusion, feel free to ask on the forum.

- **python-sat**[5]. pysat is a library that allows calls of different SAT solvers. In this project, the interface with pysat is already fixed in `exam_scheduler.ipynb`. You simply need to install it using pip.

Remark. *In principle, the Jupyter notebook should install the required libraries. But if you have to install it manually, beware that some Linux distributions do not support the* `pip` *package management tool. In this case, you can create a new local Python environment using VS Code or any other tool. This will create a* `.venv` *folder. You can install the libraries locally using the* `.venv/bin/pip3` *executable.*

Deliverables. If you solve project problem 2, you are asked to deliver two files.

1. `proj2.*`: A description of your encoding and how you solved the problem. Evaluate how your approach scales with the number of students and exams. Estimate the number of variables and clauses that your encoding generates depending on the number of students, days, courses, and rooms. This document should not be more than two pages long.

2. `exam_scheduler.py`: Your code for encoding and decoding the exam scheduling problem.

Impress me. If you wish to impress the teaching team, you may (optionally) discuss how an impossible scheduling task may be similar to the pigeon hole problem (famously difficult for SAT solver). You may explain a way to make the encoding better suited for SAT solvers in a `proj2.*` file. If you complete this part of the project, the page limit is extended to 3.

[4] `https://pandas.pydata.org/docs/`
[5] `https://pysathq.github.io/`