Assignment 2

Problem 1
Consider a formula \( \phi = \omega_1 \land \cdots \land \omega_6 \) where
\[
\begin{align*}
\omega_1 &= (x_1 \lor x_3 \lor x_4)
\omega_2 &= (x_1 \lor x_3 \lor \neg x_4)
\omega_3 &= (x_1 \lor \neg x_3 \lor x_4)
\omega_4 &= (x_1 \lor \neg x_3 \lor \neg x_4)
\omega_5 &= (\neg x_1 \lor x_2)
\omega_6 &= (\neg x_1 \lor \neg x_2)
\end{align*}
\]

**Part A.** The result of resolving a clause \((x_1 \lor \cdots \lor x_n \lor r)\) with another clause \((\neg r \lor y_1 \lor \cdots \lor y_m)\) on \(r\) is the disjunction of the literals in the set \(\{x_1, \ldots, x_n\} \cup \{y_1, \ldots, y_m\}\). Note that the resulting clause is logically implied by the conjunction of the two original clauses.

1. Let \(\omega_{10}\) be the result of resolving \(\omega_1\) with \(\omega_2\) on \(x_4\). What is \(\omega_{10}\)?
2. Let \(\omega_{11}\) be the result of resolving \(\omega_3\) with \(\omega_4\) on \(x_4\). What is \(\omega_{11}\)?
3. (a) On what variable can \(\omega_{10}\) and \(\omega_{11}\) be resolved?
   (b) Let \(\omega_{12}\) be the result of this resolution. What is \(\omega_{12}\)?
4. Let \(\omega_{13}\) be the result of resolving \(\omega_5\) and \(\omega_6\) on an appropriate literal. What is \(\omega_{13}\)?
5. Let \(\omega_{14}\) be the result of resolving \(\omega_{12}\) and \(\omega_{13}\) on an appropriate literal. What is \(\omega_{14}\)?
6. What does the above tell you about the satisfiability of \(\phi\)? Briefly explain in one or two sentences.

**Part B.** Consider a SAT solver as described in the GRASP lecture slides, in particular the slides titled “Top-level of GRASP-like solver” and “Learning Algorithm”. Suppose that the decision heuristic is to pick the lowest-numbered unassigned variable and assign it the value false. Consider executing this SAT solver on the formula \(\phi = \omega_1 \land \cdots \land \omega_6\). For every conflict encountered, give:

1. the implication graph (using the format given on the slide titled “Implication Graphs”),
2. the initial conflict assignment (from Step 1 on the “Learning Algorithm” slide),
3. the final conflict assignment (from Step 4 of the slide),
4. the learned clause, and
5. the decision level to which to backtrack (if the learned clause is non-empty).

Include decision levels in your conflict assignments; use the format “\(\{x_5=1 \oplus 6, x_6=1 \oplus 6\}\)”.

1
Problem 2: Pigeonhole Problem

Consider the problem of placing \( n \) pigeons in \( m \) pigeonholes. If \( n > m \), then at least one pigeonhole must contain more than one pigeon.

Given \( n \) pigeons and \( m \) pigeonholes, we wish to write a CNF formula \( \phi \) such that \( \phi \) is satisfiable iff each pigeon can be put in some pigeonhole and each pigeonhole has at most one pigeon. It turns out that a direct encoding of this pigeonhole problem is quite difficult for SAT solvers when \( n > m \).

Let \( z_{p,h} \) be a propositional variable which is true when pigeon \( p \) is placed in hole \( h \).

Notation: If \( S \) is a set of clauses, we’ll write “\( \bigwedge S \)” to denote the conjunction of these clauses. For example, \( (\bigwedge \{C_1, C_2, C_3\}) = (C_1 \land C_2 \land C_3) \).

Let \( E \) be a conjunction of clauses which encode the requirement that each pigeon must be in some hole, defined as follows:

\[
E = \bigwedge \left\{ (z_{p,1} \lor z_{p,2} \lor \ldots \lor z_{p,m}) \mid 1 \leq p \leq n \right\}
\]

\[
= (z_{1,1} \lor z_{1,2} \lor \ldots \lor z_{1,m}) \land (z_{2,1} \lor z_{2,2} \lor \ldots \lor z_{2,m}) \land \ldots \land (z_{n,1} \lor z_{n,2} \lor \ldots \lor z_{n,m})
\]

Let \( H_h \) be a conjunction of clauses which encode the requirement that hole \( h \) cannot have more than one pigeon, defined as follows:

\[
H_h = \bigwedge \left\{ (\neg z_{i,h} \lor \neg z_{j,h}) \mid 1 \leq i < j \leq n \right\}
\]

The final formula is

\[
\phi = E \land H_1 \land H_2 \land \ldots \land H_m
\]

To give this problem to a SAT solver that needs integer-numbered variables, we renumber as follows: \( z_{p,h} \) becomes \( x_{p \times m + h} \).

Consider the SAT solver given on Slide 13 (“DPLL Solver”) of the Lecture 2 slides. Assume the decision heuristic is as follows: Pick the first literal in the first unresolved clause and assign it true.

1. For the case of 4 pigeons and 3 holes, draw the decision tree followed by the SAT solver. (Only include decision literals, not forced literals.) A conflict should be indicated by a special leaf node, as in Slide 21. (You may use a simple star to indicate a conflict leaf node.)

2. Given \( m \) pigeonholes and \( m + 1 \) pigeons, how many conflicts would occur, as a function of \( m \)? (Each time the line “if [] in ClauseList: return False” returns false counts as one conflict.) Explain your reasoning.
Problem 3: MiniSAT
Most fast SAT solvers use the DIMACS input format. A CNF formula $\phi = C_1 \land \ldots \land C_m$, where $C_i = \ell_{i,1} \land \ldots \land \ell_{i,n_i}$, is encoded in DIMACS as follows:

```
p cnf NumVars NumClauses
\ell_{1,1} \ell_{1,2} \ldots \ell_{1,n_1} 0
\ell_{2,1} \ell_{2,2} \ldots \ell_{2,n_2} 0
\ldots
\ell_{m,1} \ell_{m,2} \ldots \ell_{m,n_m} 0
```

Each clause is described in a line terminated by a zero. As an example, the formula $(x_1) \land (x_2 \lor \neg x_3) \land (\neg x_4 \lor \neg x_1) \land (\neg x_1 \lor \neg x_2 \lor x_3 \lor x_4) \land (\neg x_2 \lor x_4)$ would be encoded as:

```
p cnf 4 5
1 0
2 -3 0
-4 -1 0
-1 -2 3 4 0
-2 4 0
```

For this assignment we will use the MiniSat SAT solver, which is one of the fastest SAT solvers currently available.

You can find a binary executable of the MiniSat solver for Linux in

```
/afs/andrew.cmu.edu/usr16/wkleiber/public/
```

You can find DIMACS files for the pigeonhole problem at

```
http://www.cl.cam.ac.uk/~tw333/software/pigeonhole/
```

Run MiniSAT on pigeon-3.cnf through pigeon-8.cnf (inclusive) and report the number of conflicts for each file. (The numbers will not be exactly the same as the numbers given by the formula in your answer to Problem 2, but they shouldn’t differ by more than a factor of 2.)