Today’s exercises

- 3.1: Decision vs. Construction
- 3.5: Keeping the number of satisfying assignments
- 3.6: Undetectedly unsatisfiable
- 3.8: Proof of Lemma 3.3
- 3.9: Infinite return time
- 3.12: Worst-Case Choice Rule
- 3.15: Dependency Bounds for 2-CNF
3.1: Decision vs. Construction

Suppose $F$ is satisfiable. We build a satisfying assignment $\alpha$ as follows:

- Check if $F[x \mapsto 0]$ and if $F[x \mapsto 1]$ is satisfiable (at least one of them must be).

- Assign $x$ s.t. the resulting formula is satisfiable, then repeat with $F \leftarrow F[x \mapsto \alpha(x)]$.

Note that this does not work if we want to count the number of satisfying assignments. (2-SAT can be decided in poly-time, but counting the number of assignments is NP-hard)
3.5: Keeping the number of satisfying assignments

If \( \{u\} \in F \), then \( |\text{sat}_V(F')| = |\text{sat}_V\backslash\text{vbl}(u)(F[u\mapsto 1])| \).

If \( F \) contains 2-clauses over the same variables, we do the following:

- If there are four 2-clauses, then \( |\text{sat}_V(F')| = 0 \). If there are three 2-clauses, then the values of the two variables are determined.

- If there are 2-clauses \( \{u,v\}, \{u,\bar{v}\} \), then \( u \mapsto 1 \) in all satisfying assignments.

- If there are 2-clauses \( \{u,v\}, \{\bar{u},\bar{v}\} \), then \( u \) and \( v \) have the different values in all satisfying assignments.

Replace all occurrences of \( v \) by \( \bar{u} \) and all occurrences of \( \bar{v} \) by \( u \). Remove clauses that are trivially satisfied.
3.6: Undetectedly unsatisfiable

Consider

\[ F = \{\{\overline{x}, y, z\}, \{\overline{x}, \overline{y}, z\}, \{\overline{x}, y, \overline{z}\}, \{\overline{x}, \overline{y}, \overline{z}\}\}, \]

which is clearly satisfiable by letting \( x \mapsto 0 \).

We have that \( F[x \mapsto 1] \) is the complete 2-CNF formula on variables \( \{y, z\} \). There is no unit clause and so unit clause reduction will return \( F \) with \( \square \not\in F \).
3.8: Proof of Lemma 3.3

We need to show two things:

1. All $e_s := E(\min\{t \mid W_t \in \{a, b\}\}|W_0 = s)$ are finite.

   No matter where we are (but not at $b$ because then we’d be done), in at most $b-a-1$ steps we hit $a$ with probability $\left(\frac{1}{2}\right)^{b-a-1}$. Hence

   $$e_s \leq b-a-1 + (b-a-1) \sum_{i=1}^{\infty} \left(1 - \left(\frac{1}{2}\right)^{b-a-1}\right)^i < \infty$$
2. The system of equations (3.1) has a unique solution.

We have a system of \( n \) linear equations and \( n \) variables. It has a unique solution iff its matrix is invertible i.e. when its determinant is nonzero. Let \( D_n \) denote the value of the determinant for the case \( n = b - a + 1 \).

Then one establishes (blackboard) the recurrence \( D_n = D_{n-1} - \frac{1}{4}D_{n-2} \) with initial values \( D_2 = 1 \), \( D_2 = 1 \). The solution to the recurrence (induction!) is \( D_n = 2^{2-n}(n - 1) > 0 \).
3.9: Infinite return time

Observe that after one step we are either at $-1$ or $1$. By symmetry it is enough to observe that happens when we start at $1$.

Let $p_i := \Pr[\exists t \in \mathbb{N}_0 : W_t = 0 | W_0 = i]$. 

We have $p_0 = 1$ and for all $i \neq 0 : p_i = \frac{1}{2}p_{i-1} + \frac{1}{2}p_{i+1}$.

Since every $p_i$ is the average of the two adjacent $p_i$, we have that $i \mapsto p_i$ must be an affine function.

The only affine function satisfying $p_0 = 1$ and $p_{-1} = p_1$ is $p_i = 1 : \forall i$. 

Algorithms for 2-SAT

Jerri Nummenpalo
3.9: Infinite return time (2)

Consider a symmetric random walk \((W_t)_{t \in \mathbb{N}_0}\) starting at 1. If we reach 0, we remain at 0.

Let \(T := \min\{t \in \mathbb{N}_0 \mid W_t = 0\}\). We want to show that \(E[T]\) must be infinite.
We have $E[W_t] = E[W_{t-1}] = \cdots = E[W_0] = 1$. Intuitively, this is because at every step we go the left and right with the same probability.

Formally (thanks to the student providing that proof),

$$E[W_t] = E \left[ W_0 + \sum_{i=1}^{t} (W_i - W_{i-1}) \right]$$

$$= E[W_0] + \sum_{i=1}^{t} (E[W_i - W_{i-1}])$$

$$= E[W_0] + \sum_{i=1}^{t} \left( \Pr[W_{i-1} = 0] \cdot 0 + \Pr[W_{i-1} \neq 0] \cdot \left( \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot (-1) \right) \right)$$

$$= E[W_0] = 1.$$
Furthermore $0 \leq W_t \leq t + 1$. It follows

$$1 = E[W_t] \leq (t + 1) \Pr[T > t].$$

and so

$$\Pr[T > t] \geq E[W_t]/(t + 1) = 1/(t + 1).$$

However

$$E[T] = \sum_{t=0}^{\infty} t \Pr[T = t] = \sum_{i=1}^{\infty} \Pr[T \geq i]\]

$$= \sum_{i=1}^{\infty} \Pr[T > i - 1] \geq \sum_{i=1}^{\infty} \frac{1}{i}.$$

This is the harmonic series which is well-known to diverge.
3.12: Worst-Case Choice Rule

If we use the maximal 2-CNF formula satisfied only by the all-one assignment, but start at the all-zero assignment...

... and then in every step choose a clause over one variable currently set to one and one variable currently set to zero (which always exist, except at the starting point) ...

... then the algorithm exactly performs a random walk reflecting at $n$ and starting at $s = n$, for which we know the expected time until arrival at zero is exactly $s(2s - n) = n^2$. 
3.15: Dependency Bounds for 2-CNF

Local Lemma yields: \( b \geq 2^{k-2} = 1 \)

Easy to see: \( b < 3 \), since the smallest unsatisfiable 2-CNF is \( \{\{x, y\}, \{\bar{x}, y\}, \{x, \bar{y}\}, \{\bar{x}, \bar{y}\}\} \), and every clause has three neighbors.

Remaining question: what about \( b = 2 \)?
3.15: Dependency Bounds for 2-CNF (2)

If $b = 2$, consider the components of the dependency graph (the vertices are clauses). The components are independent of each other. For a component, depending on the maximal number of clauses over the same variable set, do the following:

- 3 clauses over the same variable set: these clauses must be independent from all other and can be satisfied.
- 2 clauses over the same variable set: these clauses can have at most one common neighbor; and this neighbor cannot have other neighbors. Here we can match to each clause a variable.
- Otherwise the component is either a path or a cycle where every adjacent clauses share exactly one variable. We can match to each clause a variable.