1 Satisfiability

A CNF formula over boolean variables is a conjunction (AND) of clauses, where each clause is a disjunction (OR) of literals (variables or their negation). A $k$-CNF is a CNF formula where each clause contains exactly $k$ literals. So for example, the following is a 3-CNF

$$\varphi(x_1, \ldots, x_6) = (x_1 \lor \neg x_2 \lor x_3) \land (x_1 \lor \neg x_3 \lor x_5) \land (\neg x_1 \lor \neg x_2 \lor x_4) \land (x_1 \lor \neg x_2 \lor x_6).$$

The $k$-SAT problem is the computational problem of deciding whether a given $k$-CNF has a satisfying assignment. It is known to be solvable in polynomial time for $k = 2$, but it is NP-hard for $k \geq 3$, and the best algorithms run in time $\exp(c_k n)$ for some constant $0 < c_k < 1$. We will see two randomized algorithms: one that solves 2-SAT in expected polynomial time; and one that solves 3-SAT in time $2^{cn}$ for some $c < 1$. So, it improves upon enumeration of all possible solutions, which would take time $2^n$.

2 2-SAT

Let $x = (x_1, \ldots, x_n)$ and let $\varphi(x)$ be a 2-SAT given by

$$\varphi(x) = C_1(x) \land \ldots \land C_m(x),$$

where each $C_i$ is the OR of two literals. We say that an assignment $x \in \{0,1\}^n$ satisfies a clause $C_i$ if $C_i(x) = 1$. We will analyze the following simple looking algorithm.
Function Solve-2SAT

Input: 2-CNF $\phi$
Output: $x \in \{0, 1\}^n$ such that $\phi(x) = 1$

1. Set $x = 0$.
2. While there exists some clause $C_i$ such that $C_i(x) = 0$:
   2.1 Let $x_a, x_b$ be the two variables participating in $C_i$.
   2.2 Choose $\ell \in \{a, b\}$ uniformly: $\Pr[\ell = a] = \Pr[\ell = b] = 1/2$.
   2.3 Flip $x_\ell$.
3. Output $x$.

We will show the following theorem.

**Theorem 2.1.** If $\phi$ is a satisfiable 2-CNF, then with probability 99%, the algorithm will find a solution in time at most $100n^2$.

For the proof, fix some solution $x^*$ for $\phi$ (if there is more than one, choose one arbitrarily). Let $x^t$ denote the value of $x$ in the $t$-th iteration of the loop. Note that it is a random variable, which depends on our choice of which variables to flip in all the previous steps. Define $d_t = \text{dist}(x^t, x^*)$. Clearly, at any stage $0 \leq d_t \leq n$, and if $d_t = 0$ then we output $x_t = x^*$ at iteration $t$.

Consider $x^t$, the assignment at iteration $t$, and assume that $d_t > 0$. Let $C = \ell_a \lor \ell_b$ be a violated clause, where $\ell_a \in \{x_a, \neg x_a\}$ and $\ell_b \in \{x_b, \neg x_b\}$. This means that either $(x^*_a) \neq (x^t)_a$ or $(x^*_b) \neq (x^t)_b$ (or both), since $C(x^*) = 1$ but $C(x^t) = 0$. If we choose $\ell \in \{a, b\}$ such that $(x^t)_\ell \neq (x^*_\ell)$, then the distance between $x^{t+1}$ and $x^*$ decreases by one; otherwise, it increases by one. So we have:

$$d_{t+1} = d_t + \Delta_t,$$

where $\Delta_t \in \{-1, 1\}$ is a random variable that satisfies $\Pr[\Delta_t = -1|x^t] \geq 1/2$. That is, the sequence of steps $d_0, d_1, d_2, \ldots$ is conducting a random walk on $\{0, 1, \ldots, n\}$, with a probability of at least 1/2 of going to the left (decreasing the value). We will show that after $O(n^2)$ steps, with high probability, this has to terminate: either some satisfying assignment has been found, or otherwise we hit 0 and output $x^*$. We do so by showing that a random walk tends to drift far from its origin. We first analyze the slightly simpler case where the random walk is symmetric, that is $\Pr[\Delta_t = -1|x^t] = \Pr[\Delta_t = 1|x^t] = 1/2$. We will then show how to extend the analysis to our case, where the probability for $-1$ could be larger.

**Lemma 2.2.** Let $y_0, y_1, \ldots$ be a random walk, defined as follows: $y_0 = 0$ and $y_{t+1} = y_t + \Delta_t$, where $\Delta_t \in \{-1, 1\}$ and $\Pr[\Delta_t = 1|y_t] = 1/2$ for all $t \geq 0$. Then, for any $t \geq 0$,

$$\mathbb{E}[y_t^2] = t.$$
Proof. We prove this by induction on $t$. It is clear for $t = 0$. We have
\[
\mathbb{E}[y_{t+1}^2] = \mathbb{E}[(y_t + \Delta_t)^2] = \mathbb{E}[y_t^2] + 2\mathbb{E}[y_t\Delta_t] + \mathbb{E}[\Delta_t^2].
\]
By induction, $\mathbb{E}[y_t^2] = t$. Since $\Delta_t \in \{-1,1\}$ we have $\mathbb{E}[\Delta_t^2] = 1$. To conclude, we need to show that $\mathbb{E}[y_t\Delta_t] = 0$. We show this via the rule of conditional expectations:
\[
\mathbb{E}[y_t\Delta_t] = \mathbb{E}_{y_t}[\mathbb{E}_{\Delta_t}[y_t\Delta_t|y_t]] = \mathbb{E}_{y_t}[y_t \cdot \mathbb{E}_{\Delta_t}[\Delta_t|y_t]] = \mathbb{E}_{y_t}[y_t \cdot 0] = 0.
\]

We now prove the more general lemma, where we allow a consistent drift.

Lemma 2.3. Let $y_0, y_1, \ldots$ be a random walk, defined as follows: $y_0 = 0$ and $y_{t+1} = y_t + \Delta_t$, where $\Delta_t \in \{-1,1\}$ and $\Pr[\Delta_t = 1|y_t] \geq 1/2$ for all $t \geq 0$. Then, for any $t \geq 0$,
\[
\mathbb{E}[y_t^2] \geq t/2.
\]

Note that the same result holds by symmetry if we assume instead that $\Pr[\Delta_t = -1|y_t] \geq 1/2$ for all $t \geq 0$

Proof. The proof is by a “coupling argument”. Define a new random walk $y'_0, y'_1, \ldots$, where $y'_0 = 0, y'_{t+1} = y'_t + \Delta'_t$. In general, we would have that $y'_t, \Delta'_t$ depend on $y_1, \ldots, y_t$. So, fix $y_1, \ldots, y_t$, and assume that $\Pr[\Delta_t = 1|y_t] = \alpha$ for some $\alpha \geq 1/2$. Define $\Delta'_t$ as:
\[
\Delta'_t(y_t, \Delta_t) = \begin{cases} 
-1 & \text{if } \Delta_t = -1 \\
1 & \text{if } \Delta_t = 1, \text{ with probability } 1/2\alpha \\
-1 & \text{if } \Delta_t = 1, \text{ with probability } 1 - 1/2\alpha 
\end{cases}
\]

It satisfies the following properties:

\begin{itemize}
\item $\Delta'_t \leq \Delta_t$.
\item $\Pr[\Delta'_t = 1|y_t, \Delta_t] = 1/2$.
\end{itemize}

Note that the random walk $y'_0, y'_1, \ldots$, is a symmetric random walk, which further satisfies $y'_t \leq y_t$ for all $t \geq 0$. By the previous lemma,
\[
\mathbb{E}[(y'_t)^2] = t.
\]

Note moreover that since the random walk $y'_0, y'_1, \ldots$ is symmetric, $\Pr[y'_t = a] = \Pr[y'_t = -a]$ for any $a \in \mathbb{Z}$. In particular, $\Pr[y'_t \geq 0] \geq 1/2$. Whenever $y'_t \geq 0$ we have $y'_t^2 \geq (y'_t)^2$. So we have
\[
\mathbb{E}[y_t^2] \geq \mathbb{E}[y'_t^2 \cdot 1_{y'_t \geq 0}] \geq \mathbb{E}[(y'_t)^2 \cdot 1_{y'_t \geq 0}] = \mathbb{E}[(y'_t)^2]/2 = t/2.
\]
We return now to the proof of theorem. Let $E$ be the event that the algorithm doesn’t terminate within the first $T = 200n^2$ steps. Let $y_i = d_0 - d_i$, so that conditioned on $E$ holding, $y_0, y_1, \ldots, y_T$ satisfy the conditions of the lemma. Then

$$\mathbb{E}[y_T^2] \geq \mathbb{E}[y_T^2 | E] \Pr[E] \geq T/2 \cdot \Pr[E].$$

However, since $0 \leq d_0, d_i \leq n$ we have that $|y_T| \leq n$. So

$$\Pr[E] \leq \frac{n^2}{T/2} \leq 1\%.$$ 

3 3-SAT

Let $\varphi$ be a 3-CNF. Finding if $\varphi$ has a satisfying assignment is NP-hard, and the best known algorithms take exponential time. However, we can still improve upon the naive $2^n$ full enumeration, as the following algorithm shows. The algorithm we analyze is due to Schoening. Let $T$ be a parameter to be determined later.

<table>
<thead>
<tr>
<th>Function Solve-3SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong>: 3–CNF $\varphi$</td>
</tr>
<tr>
<td><strong>Output</strong>: $x \in {0,1}^n$ such that $\varphi(x) = 1$</td>
</tr>
<tr>
<td>1. Choose $x \in {0,1}^n$ randomly.</td>
</tr>
<tr>
<td>2. For $i = 1, \ldots, T$:</td>
</tr>
<tr>
<td>2.1 If $\varphi(x) = 1$, output $x$.</td>
</tr>
<tr>
<td>2.2 Otherwise, let $C_i$ be some clause such that $C_i(x) = 0$, with variables $x_a, x_b, x_c$.</td>
</tr>
<tr>
<td>2.2 Choose $\ell \in {a, b, c}$ uniformly:</td>
</tr>
<tr>
<td>$\Pr[\ell = a] = \Pr[\ell = b] = \Pr[\ell = c] = 1/3$.</td>
</tr>
<tr>
<td>2.3 Flip $x_\ell$.</td>
</tr>
<tr>
<td>3. Output FAIL.</td>
</tr>
</tbody>
</table>

Our goal is to analyze the following question: what is the success probability of the algorithm? as before, assume $\varphi$ is satisfiable, and choose some satisfying assignment $x^\ast$. Define $x^i$ to be the value of $x$ at the $i$-th iteration of the algorithm, and let $d_i = \text{dist}(x^i, x^\ast)$. 

**Claim 3.1.** The following holds

1. $\Pr[d_0 = k] = 2^{−n}{n \choose k}$ for all $0 \leq k \leq n$.
2. $d_{i+1} = d_i + \Delta_i$ where $\Delta_i \in \{-1, 1\}$ satisfies $\Pr[\Delta_i = -1|d_i] \geq 1/3$.

**Proof.** For (i), note that $d_0$ is the distance of a random string from $x^\ast$, so equivalently, it is the hamming weight of a uniform element of $\{0,1\}^n$. The number of elements of hamming
weight \( k \) is \( \binom{n}{k} \). For (ii), if \( x_a, x_b, x_c \) are the variables appearing in an unsatisfied clause at iteration \( t \), then at least one of them disagrees with the value of \( x^* \). If \( w \) we happen to choose it, the distance will decrease by one, otherwise, it will increase by one.

For simplicity, lets assume from now on that \( \Pr[\Delta_t = -1|d_t] = 1/3 \), where the more general case can be handled similar to the way we handled it for 2-SAT.

**Claim 3.2.** Assume that \( d_0 = k \). The probability that the algorithm finds a satisfying solution is at least

\[
\left( \frac{T}{(T+k)/2} \right) \left( \frac{1}{3} \right)^{(T+k)/2} \left( \frac{2}{3} \right)^{(T-k)/2}.
\]

**Proof.** Consider the sequence of steps \( \Delta_0, \ldots, \Delta_{T-1} \). If there are \( k \) more \(-1\) than \(+1\) in this sequence, then starting at \( d_0 = k \), we will reach \( d_T = 0 \). The number of such sign sequences is \( \binom{T}{(T+k)/2} \); the probability for seeing a \(-1\) is a \( 1/3 \), and the probability for seeing a \(+1\) is \( 2/3 \).

**Corollary 3.3.** The probability that the algorithms finds a solution is at least

\[
2^{-n} \left( \frac{n}{k} \right) \left( \frac{T}{(T+k)/2} \right) \left( \frac{1}{3} \right)^{(T+k)/2} \left( \frac{2}{3} \right)^{(T-k)/2}.
\]

We now need to optimize parameters. Fix \( k = \alpha n, T = \beta n \). We will use the following approximation: for \( n \geq 1 \) and \( 0 < \alpha < 1 \),

\[
\left( \frac{n}{\alpha n} \right) \approx \left( \frac{1}{\alpha} \right)^{\alpha n} \left( \frac{1}{1-\alpha} \right)^{(1-\alpha)n} \approx 2^{H(\alpha)n},
\]

where \( H(\alpha) = \alpha \log_2(1/\alpha) + (1-\alpha) \log_2(1/(1-\alpha)) \). Then

\[
\left( \frac{n}{k} \right) \approx 2^{H(\alpha)n},
\]

\[
\left( \frac{T}{(T+k)/2} \right) \approx 2^{H(1/2+\alpha/2\beta)\beta n}
\]

\[
\left( \frac{1}{3} \right)^{(T+k)/2} \left( \frac{2}{3} \right)^{(T-k)/2} = 3^{-\beta n}2^{(\beta-\alpha)/2}n.
\]

So, we can express the probability of success as \( \approx 2^{-\gamma n} \), where

\[
\gamma = 1 - H(\alpha) - H(1/2 + \alpha/2\beta)\beta + \log_2 3 \cdot \beta - (\beta - \alpha)/2.
\]

Our goal is to choose \( 0 < \alpha < 1 \) and \( \beta \geq \alpha \) to minimize \( \gamma \). The minimum is obtained for \( \alpha = 1/3, \beta = 1 \), which gives \( \gamma \approx 0.41 \).

So, we have an algorithm that runs in polynomial time, and finds a satisfiable assignment with probability \( \approx 2^{-\gamma n} \). To find a satisfiable assignment with high probability, we simply repeat it \( N = 5 \cdot 2^{\gamma n} \) times. The probability it fails in all these executions is at most

\[
(1 - 2^{-\gamma n})^N \leq \exp(-2^{-\gamma n}N) \leq \exp(-5) \leq 1%.
\]

**Corollary 3.4.** We can solve 3-SAT in time \( O(2^{\gamma n}) \) for \( \gamma \approx 0.41 \).