Informally, a DFA is some kind of machine with finite memory, that processes the input string strictly from left to right. Having n states is equivalent to having approximately $\log n$ bits of memory.

Formally, a DFA over some *alphabet* Σ is a finite directed graph, with every vertex (or *state*) having exactly $|\Sigma|$ outgoing arcs: one for every symbol from Σ. One of the states is labeled as *initial*, and some subset of them is labeled as *accepting*. Automaton *accepts* some string s, if reading s along the arcs makes it go from the initial vertex to some accepting vertex.

Suppose that you want to construct a small deterministic finite automaton (DFA) that accepts some string s but rejects some other string t. What is the minimal possible number of states in such an automaton? This is, in a sense, one of the simplest possible computational problems.

Theorem 1. *If s and t have different length, than there is an automaton with $O(\log \max(|s|, |t|))$ states that separates them.*

A sketch. Compute the length of the input string modulo all numbers m that are less than $O(\log \max(|s|, |t|))$. For one of them, the remainders will not be equal. □

From now on, we consider only the case $|s| = |t| = n$ and the alphabet Σ is binary. Larger alphabets can be encoded in binary by using $\lceil \log_2 |\Sigma| \rceil$ bits.

Simple counting heuristics suggest that the minimal separating DFA should have $O(\log n)$ states. Moreover, numeric evidence for small n suggest the same upper bound as well. However, all simple constructions fail miserably. For example, there are two string of length n, such that *all* polynomial hashes with modulo being at most $O(\log n)$ coincide on such strings.

It is known that there exists a separating automaton with $O(n^{2/5} \log^{3/5} n)$ states, but the proof is very technical and involves a lot of case-checking. Hence, I recited a more beautiful proof of slighlty weaker result:

Theorem 2. *For different strings s and t of length n over binary alphabet, there is a separating DFA with $O(\sqrt{n})$ states.*

A sketch. Consider the function $f_{m,x}(w)$ that computes the parity of number of 1 on the positions that have remainder x modulo m. This function can be computed with an automaton with $O(m)$ states.

As it turns out, for every two strings s and t, there is always $m = O(\sqrt{n})$ and $x \in [0, m)$, such that $f_{x,m}(s) \neq f_{x,m}(t)$. Hence, there is a separating automaton with $O(\sqrt{n})$ states. The proof is pretty cool, but I will omit it, because Hirsh most probably won't ask anything about it. If you are interested, you can see the original paper (J. M. Robson, "Separating words with machines and groups"). □