1 Lecture summary

- We proved that DFA’s find it hard to count (as proposed in the last lecture)
- We generalized the proof to state and prove the pumping lemma
- We used the pumping lemma to prove that \( \{0^n1^n \mid n \in \mathbb{N} \} \) is not DFA-
  recognizable

2 DFAs find it hard to count

We claim that any machine that recognizes the language \( \{1^c\} \) must have at least
\( c \) states.

Important note: the language \( \{1^c\} \) is different from \( \{1^c \mid c \in \mathbb{N} \} \).

Important note: a machine only recognizes a language \( L \) if

- it says yes on all inputs in \( L \)
- AND: it says no on all inputs not in \( L \)

In particular, although the machine that recognizes all strings has only one state, and does “recognize” every string in \( 1^c \), it does not “recognize” \( \{1^c\} \), because it also accepts other strings (such as \( 1^{(c+1)} \))

Proof of claim: Proof by contradiction. Suppose that \( M \) recognizes \( L \) and \( M \) has fewer than \( c \) states. While processing the string \( 1^c \), \( M \) passes through states \( q_0, q_1, q_2, \ldots, q_c \). There are \( (c + 1) \) such states, but there are fewer than \( c \) states in \( M \), so the same state must be repeated twice in the sequence, i.e. \( q_i = q_j \) for some \( i \) and \( j \).

This means there is a loop; if we add an extra \( (j - i) \) ‘1’s to the string, it will still be accepted, it will just traverse the loop an extra time. Therefore \( 1^{(c+(j-i))} \) is in the language of \( M \), which contradicts the fact that \( L(M) = \{1^c\} \)

Therefore, there is no machine having fewer than \( c \) states that recognizes \( \{1^c\} \).

3 The pumping lemma

We can use the same kind of proof technique to prove that certain languages
cannot be recognized by \textit{any} machine. The main tool for doing this is called
the pumping lemma.
Claim (pumping lemma): If $M$ is a DFA with $n$ states, and $x \in L(M)$, and $|x| > n$, then there exist strings $u$, $v$, and $w$ such that

1. $uvw = x$
2. $|v| \geq 1$
3. $|uv| \leq n$
4. for all $c$, $uv^cw$ is in $L(M)$

Proof is below.

4 Example using the pumping lemma

Claim: the language \{0^n1^n|n \in \mathbb{N}\} is not DFA-recognizable.

Proof: by contradiction. Suppose there exists a DFA $M$ that recognizes $L$. Let $k$ be the number of states of $M$. Since $L = L(M)$, the string $0^k1^k$ is recognized by $M$. Since $|0^k1^k| > k$, we can apply the pumping lemma to find some $u$, $v$, and $w$ such that $0^k1^k = uvw$, and satisfying the other properties given by the pumping lemma.

Since $|vw| \leq k$, we know that $v$ must only contain ‘0’s. Therefore, if we pump $v$ up, we have $uv^2w = 0^{k+|v|}1^k$, which we are guaranteed is in $L(M)$. But this string is not in $L$, since it has more ‘0’s than ‘1’s. This contradicts the assumption that $L = L(M)$, and concludes the proof of the claim.

5 Proof of pumping lemma

Consider the first $n + 1$ states traversed while $M$ processes $x$: $q_0$, $q_1$, \ldots, $q_n$.

Since there are $n + 1$ of them, and $M$ has only $n$ states, we must have $q_i = q_j$ for some $i \neq j$.

Let

• $u$ be the first $i$ characters of $x$.
• $v$ be the next $(j - i)$ characters of $x$.
• $w$ be the last $(|x| - j)$ characters of $x$.

Then clearly $x = uvw$.

Moreover, $|v| \geq 1$ since $j \neq i$.

In addition, $|uw| \leq n$ since $|uv| = j \leq n$.

Finally, while processing $uv^cw$, $M$ will traverse the states

$$q_0 q_1 \cdots q_{i-1} \overbrace{q_i \cdots (q_j = q_i) \cdots (q_j = q_i) \cdots q_j}^{c \text{ times}} q_{j+1} \cdots q_{|x|}$$

and will therefore end up in $q_{|x|}$. Since $x$ was accepted, $q_{|x|}$ must be an accepting state, so $uv^cw$ will be accepted.