COMS W3261: Computer Science Theory, Spring 2017.
Instructor: Tal Malkin

DFA can be exponentially larger than equivalent NFA

These notes cover a theorem we proved in lecture 4, which is not covered in Sipser's textbook. The theorem shows that sometimes the transformation from an NFA to an equivalent DFA necessitates an exponential blow up in the number of states. We show this by exemplifying a specific language for which we prove that the best possible NFA is exponentially smaller than the best possible DFA.

Definition 1. *For an arbitrary $n \in \mathbb{N}$, define the language*

$$L_n = \{w \in \{0,1\}^* \mid \textit{ the } n\textit{-th to last symbol in } w \textit{ is } 1\}$$

Claim 2. *For all n, there is an NFA with $n+1$ states recognizing L_n.*

Proof. For any fixed n, define the NFA $N_n = (\{q_0, \ldots, q_n\}, \{0,1\}, \delta, q_0, \{q_n\})$ where δ is defined as follows:

$$\begin{array}{ll} \delta(q_0, 0) = \{q_0\} & \\ \delta(q_0, 1) = \{q_0, q_1\} & \\ \delta(q_i, a) = \{q_{i+1}\} & \forall i \in \{1, \ldots, n-1\}, a \in \{0,1\} \\ \delta(q_n, a) = \emptyset & \forall a \in \{0,1\} \end{array}$$

It is not hard to verify that N_n recognizes the language L_n. □

Theorem 3. *For any $n \in \mathbb{N}$, any DFA recognizing the language L_n must have at least 2^n states.*

Proof. Assume towards contradiction that there's a DFA D with fewer than 2^n states recognizing the language L_n. Consider the set of all n bit strings. There are 2^n different n-bit strings, but fewer than 2^n states in D, so by pigeonhole principle, there must be two different strings $w, w' \in \{0,1\}^n$ such that the computation of D on w and w' ends in the same state. Note that this also means that the computation of D on wz and on $w'z$ ends in the same state, for any string z. Write $w = w_1 \ldots w_n$ and $w' = w'_1 \ldots w'_n$ (bit representation). Since $w \neq w'$, they must differ in at least one location. Take some $i \in \{1, \ldots, n\}$ such that $w_i \neq w'_i$, which means one of w_i, w'_i is 0 and the other one is 1. Append the string $z = 0^{i-1}$ to each of w, w'. The n-th to last bit in $w_1 \ldots w_n 0^{i-1}$ is w_i, and the n-th to last bit in $w'_1 \ldots w'_n 0^{i-1}$ is w'_i, one of which is 0 and one of which is 1. Thus, one of these strings is in L_n and one is not in L_n. However, the computation of the DFA D on both these strings ends in the same state, which, whether it is an accepting state or not, gives the wrong output on one of these strings. That is, one of these two strings $w0^{i-1}$ or $w'0^{i-1}$ is a counter example contradicting the fact that D recognizes L_n. □

We have just proved that there exists a language (L_n) where the number of states in the best possible DFA for the language is at least 2^n, but there is an NFA for the language with $n+1$ states. This implies that the exponential blow up in the number of states we have in our subset construction is inherent, and this aspect cannot be improved. That is:

Corollary 4. *Any algorithm that can transform any NFA with m states to an equivalent DFA with $f(m)$ states, must have $f(m) = \Omega(2^m)$*

Remarks

- We note that this does not mean that for every language the best NFA is smaller than the best DFA (we saw several examples where this was not the case; one simple example is the one-state DFA for the language Σ^*, which clearly cannot be improved even if we allow an NFA).
- As a sanity check, and to make sure you understand the proof, can you see where the proof of Theorem 3 uses the fact that D is a DFA and not an NFA?
- A fundamental result in the theory of regular languages, the Myhill-Nerode theorem, characterizes the regular languages and can be used to find the minimal DFA which recognizes a given language, or to prove that the language is not regular and no DFA exists.[1] What we proved here for the language L_n can be presented in terms of the Myhill-Nerode theorem, showing that the minimal DFA for this language has size at least 2^n. It can also be shown (via the Myhill-Nerode theorem or via a construction) that there is a DFA with 2^n states for this language, and so the minimal DFA has exactly 2^n states. The Myhill-Nerode theorem is not part of the required material for our class, but it is covered briefly in the textbook in problems 1.51,1.52, and we provide a handout with further exposition and examples, courtesy of TA Michael Tong, for interested students.

[1] We also mention that given a specific DFA for a language, there are algorithms for minimizing it to get an equivalent DFA with a minimal number of states. We will not cover this in class.