Minimização de autômatos

Vimos na aula anterior que o processo de desenho/criação de AFDs envolve, de alguma forma, a criatividade do projetista. Nesse sentido, o processo é similar ao da construção de programas de computador, em que podem existir distintos programas para resolver um mesmo problema. Isso nos leva a definir quando dois AFDs são equivalentes.

☆ Dois AFDs, $M_1 e \quad M_2$, são equivalentes se, e somente se, $L(M_1) = L(M_2)$.

Essa definição, por sua vez, nos faz refletir se não existiria um autômato mínimo para uma dada linguagem. Considerando que o alfabeto só possua símbolos efetivamente úteis, os fatores que influenciam no tamanho do autômato são o número de estados e de transições. Como a função de transição é total e o alfabeto é mínimo, o número de transições depende exclusivamente do número de estados. Portanto, um AFD é **mínimo** se ele possui o menor número de estados possível para reconhecer a linguagem. Ou seja, ele é mínimo se todos os seus estados são úteis.

Antes de definir formalmente o conceito de estado útil, formalizaremos o conceito de estado de erro (discutido na aula anterior) e o conceito de estado alcançável.

☆ Um **estado de erro** e é um estado de um autômato $M = (Q, \Sigma, \delta, i, F)$ tal que a computação de qualquer palavra a partir desse estado não é reconhecida pelo autômato; ou seja, $\forall w \in \Sigma^* \quad \hat{\delta}(e, w) \notin F$.

Logo, tem-se que:

- Se $w = xy$ e $\hat{\delta}(i, x) = e$, então $w \notin L(M)$ para qualquer $y \in \Sigma^*$
- Se i é um estado de erro, então $L(M) = \emptyset$. Consequentemente, todos os estados podem ser excluídos exceto o inicial, sendo todas as transições substituídas para transições sob ele mesmo.
- Se $X \subseteq Q$ é um conjunto de estados de erro, então M pode ser minimizado, substituindo-se todos os estados de X por um único estado de erro.

☆ Um estado $e \in Q$ de um autômato $M = (Q, \Sigma, \delta, i, F)$ é chamado de **alcançável** se ele é atingido durante a computação de alguma palavra. Formalmente, $e \in Q$ é alcançável se $\exists w \in \Sigma^* \quad \hat{\delta}(i, w) = e$.

Pela definição, tem-se que:

- Todo estado alcançável é um estado útil
- Um estado alcançável pode ser de erro ou não
- $L(M) = \emptyset$ se $F = \emptyset$ ou todo estado final é inalcançável
- Todos os estados inalcançáveis podem ser eliminados do autômato, bem como as transições para esses estados. O autômato resultante certamente será menor que o original.

Dessa forma, podemos minimizar um autômato removendo estados inalcançáveis e substituindo um conjunto de estados de erro por um único estado.

O último passo para minimização completa de um autômato é a substituição de estados equivalentes por um único estado, tal como fizemos com o estado de erro. Dois estados são **equivalentes** se toda palavra for reconhecida a partir de ambos os estados. Formalmente, para um AFD $M = (Q, \Sigma, \delta, i, F)$, dois estados $e, \quad e' \in Q$ são equivalentes, denotado por $e \approx e'$, se, e somente se, $\hat{\delta}(e, y) \in F \Leftrightarrow \hat{\delta}(e', y) \in F$ para todo $y \in \Sigma^*$.

Note que, pela definição, se um sufixo é reconhecido ao atingir um estado e então ele também será reconhecido se qualquer estado $e' \approx e$ for atingido durante o processamento do prefixo. Logo, tanto faz atingir e quanto e', ambos levarão ao reconhecimento da palavra. Isso, por sua vez, justifica a manutenção de somente um dos dois estados, realizando os ajustes necessários. De fato, toda a classe de equivalência de e pode ser substituída por um único estado.

Exemplo: Considere o AFD representado pelo diagrama abaixo. Os estados finais 4 e 5 são equivalentes. Note que qualquer **<u>sufixo</u>** reconhecido passando pelo estado 4 é também reconhecido passando pelo estado 5. Vale ressaltar que <u>não estamos dizendo que o sufixo será reconhecido exatamente naquele estado, mas que será reconhecido se qualquer dos dois estados for atingido</u>.

G)

O problema agora se resume a encontrar tais classes de equivalência para que os estados possam ser substituídos. Para isso, utilizaremos uma definição indutiva para estados i-equivalentes.

Sejam e e e' dois estados de um autômato $M = (Q, \Sigma, \delta, i, F)$. Os estados são i-equivalentes, para $i \geq 0$, denotado por $e \approx_i e'$, se:

A. $e \approx_0 e'$ se, e somente se, $e \in F \Leftrightarrow e' \in F$
B. $e \approx_{i+1} e'$ se, e somente se, $e \approx_i e'$ e $\delta(e, a) \approx_i \delta(e', a)$ para todo $a \in \Sigma$

Veja que as classes de equivalência são obtidas por refinamentos sucessivos das classes anteriores. Inicia-se com uma divisão simples entre estados finais e não-finais, e refinam-se sucessivamente essas classes de equivalência obtendo novas classes que mantém os estados equivalentes com palavras de tamanho i+1. Eventualmente, um ponto fixo é obtido e os refinamentos podem ser interrompidos. Isso implica que existe um valor de i para o qual $e \approx e' \Leftrightarrow e \approx_i e'$.

A demonstração do parágrafo anterior se encontra no livro texto (Teorema 1, p. 50). A formalização do algoritmo é apresentada na Figura 2.10 na página 51.

Exemplo: Vamos minimizar o autômato do diagrama acima.

G)