3.6 Equivalência entre Conjuntos Regulares e Autômatos Finitos

A presente seção examina a equivalência entre as gramáticas lineares à direita e os autômatos finitos, quando considerados como alternativas para a representação de linguagens regulares. Tais equivalências estão representadas com destaque na Figura 3.31.

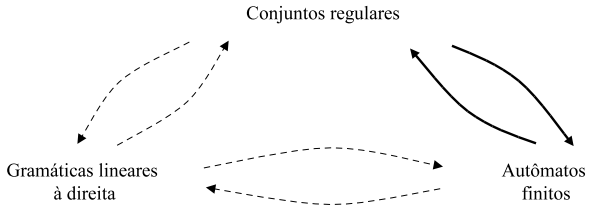

Figura 3.31. Equivalência dos formalismos — parte 3

Apresenta-se, em seguida, um algoritmo que permite a construção sistemática de autômatos finitos não-determinísticos diretamente a partir de expressões regulares quaisquer.

Teorema 3.11 (Expressões regulares ⇒ autômatos finitos) *Seja r uma expressão regular sobre o alfabeto Σ. Então existe um autômato finito M que aceita a linguagem definida por r.*

Justificativa O autômato finito não-determinístico que aceita a linguagem definida por r pode ser obtido através da aplicação do Algoritmo 3.16, que especifica as regras de mapeamento parciais que abrangem casos triviais de sentenças (itens 1, 2 e 3) e cada um dos operadores de união (4), concatenação (5) e fechamento (6), conforme a própria definição das expressões regulares.

Algoritmo 3.16 (Expressão regular ⇒ autômato) *Obtenção de um autômato finito a partir de uma expressão regular.*

- Entrada: uma expressão regular r sobre um alfabeto Σ;
- Saída: um autômato finito M tal que $L(M) = r$;
- Método:

1. $r = \epsilon$

   r é aceita por:

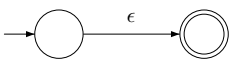

Figura 3.32. Autômato que aceita ϵ

2. $r = \varnothing$

   r é aceita por:

Figura 3.33. Autômato que aceita $\varnothing$

3. $r = \sigma, \sigma \in \Sigma$

   r é aceita por:

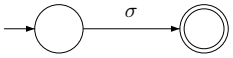

Figura 3.34. Autômato que aceita σ

4. $r_3 = r_1 \mid r_2$, com $L(M_1) = r_1$ e $L(M_2) = r_2$

   r_3 é aceita por:

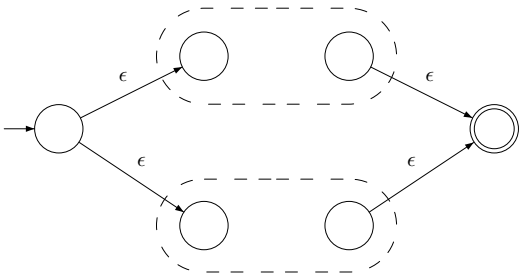

Figura 3.35. Autômato que aceita $r_1 \mid r_2$

   As cadeias aceitas por M_3 correspondem àquelas que são aceitas por M_1 e também àquelas que são aceitas por M_2. Logo, $L(M_3) = L(M_1) \cup L(M_2)$.

5. $r_3 = r_1 r_2$, com $L(M_1) = r_1$ e $L(M_2) = r_2$

   r_3 é aceita por:

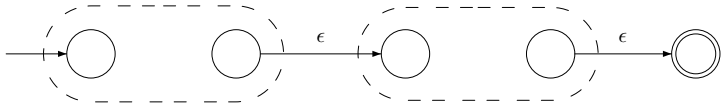

Figura 3.36. Autômato que aceita $r_1 r_2$

As cadeias aceitas por M_3 correspondem àquelas cujos prefixos sejam cadeias aceitas por M_1 e cujos sufixos sejam cadeias que sejas aceitas por M_2. Logo, $L(M_3) = L(M_1)L(M_2)$.

6. $r_3 = r_1^*$, com $L(M_1) = r_1$

   r_3 é aceita por:

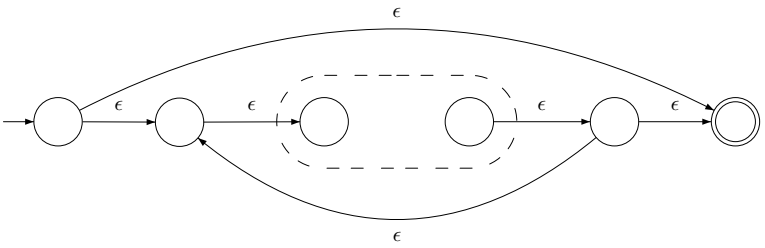

Figura 3.37. Autômato que aceita r_1^*

M_3 aceita a cadeia vazia e também a concatenação, em número arbitrário de vezes, de quaisquer cadeias que são aceitas por M_1. Logo, $L(M_3) = L(M_1)^*$.

Adicionalmente, valem as seguintes regras para os autômatos M_1 e M_2 representados nos casos (4), (5) e (6):

- Os estados originalmente finais desses autômatos tornam-se não-finais na composição e se comportam apenas como estados de saída do autômato;
- Convenciona-se que a entrada dos mesmos é pelo lado esquerdo e a saída pelo lado direito das respectivas figuras.

■

Exemplo 3.39 Considere-se a expressão regular $ab^* \mid c$. É possível identificar, nessa expressão, as seguintes linguagens triviais: $L_1 = a, L_2 = b, L_3 = c$. Portanto, $L_1 = L_1(M_1), M_1 =$

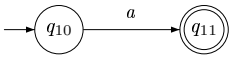

Figura 3.38. Autômato que aceita a

$L_2 = L_2(M_2), M_2 =$

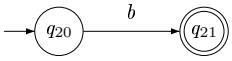

Figura 3.39. Autômato que aceita b

$L_3 = L_3(M_3), M_3 =$

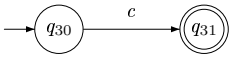

Figura 3.40. Autômato que aceita c

Seja $L_4 = b^* = L_2^*$. Então, $L_4 = L_4(M_4)$ com M_4 igual a:

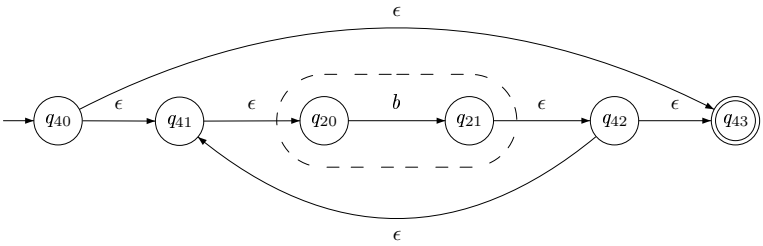

Figura 3.41. Autômato que aceita b^*

$L_5 = ab^* = L_1L_4$. Então, $L_5 = L_5(M_5)$ com M_5 igual a:

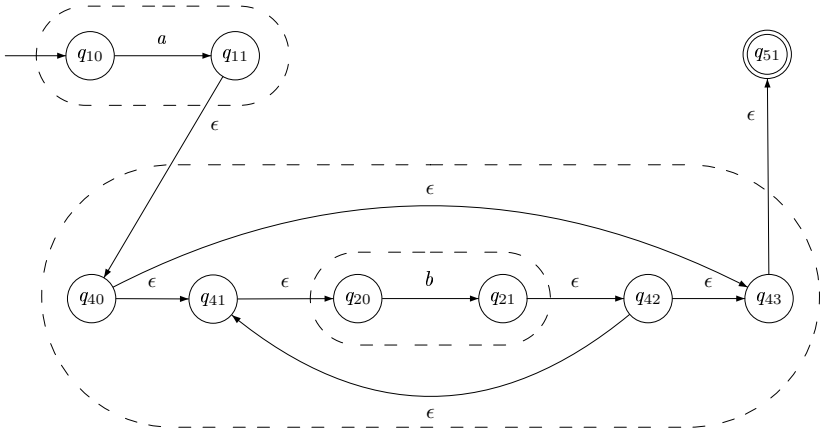

Figura 3.42. Autômato que aceita ab^*

Finalmente, $L_6 = ab^* \mid c = L_5 \cup L_3 = L_6(M_6)$ com M_6 igual a:

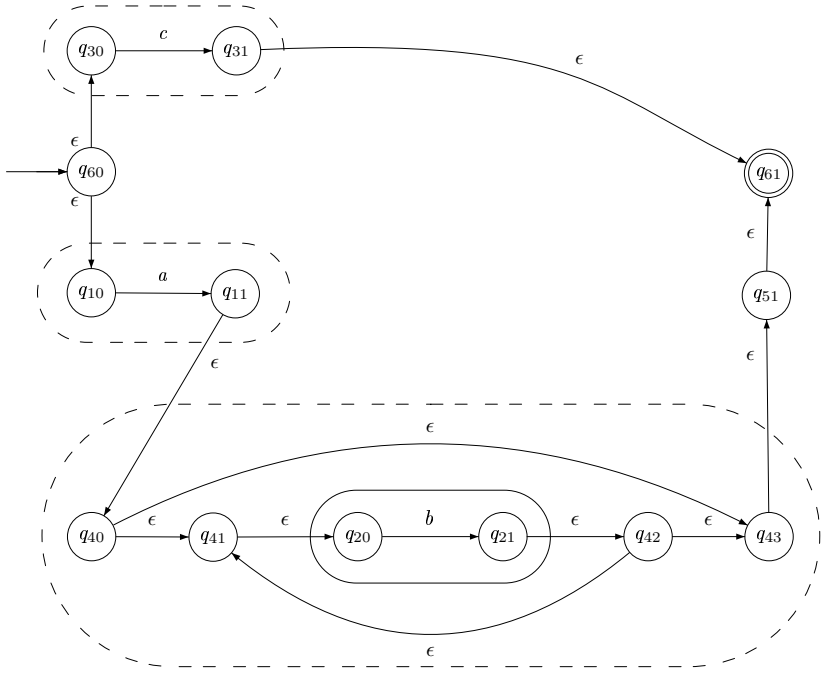

Figura 3.43. Autômato que aceita $ab^* \mid c$

Como se pode observar, o autômato assim construído apresenta uma quantidade desnecessariamente grande de estados, além de diversas transições em vazio. Este fenômeno, comum quando se aplicam métodos canônicos de construção de autômatos, pode ser contornado através da aplicação dos algoritmos de eliminação de transições em vazio e de estados inacessíveis, além da aplicação de algoritmos de minimização, assunto da Seção 3.7. O autômato da Figura 3.44 corresponde a uma versão equivalente, porém mais compacta, ao autômato M_6 anteriormente obtido.

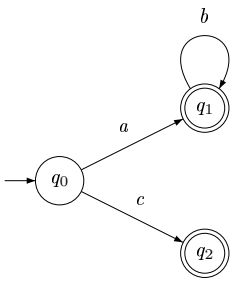

Figura 3.44. Outro autômato que aceita $ab^* \mid c$

□

É comum se considerar uma simplificação do autômato apresentado na Figura 3.37, no qual, ao invés de quatro novos estados, são criados apenas dois novos estados, como ilustrado na Figura 3.45.

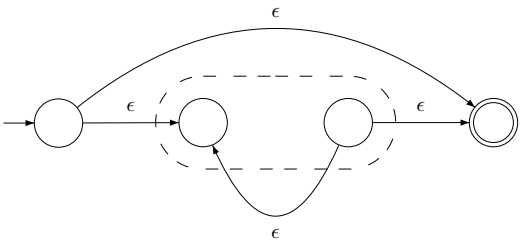

Figura 3.45. Autômato alternativo que aceita r_1^*

Assim como no caso dos autômatos apresentados nas Figuras 3.35 e 3.36, o autômato da Figura 3.37 preserva as funções de transição dos autômatos originais inalteradas. O autômato da Figura 3.45, por outro lado, exige o acréscimo de uma transição em vazio do (então) estado final para o (então) estado inicial do autômato original, porém produz resultados equivalentes aos obtidos com o autômato da Figura 3.37.

Exemplo 3.40
O autômato da Figura 3.46 reconhece a linguagem gerada pela expressão regular a^*b^*.

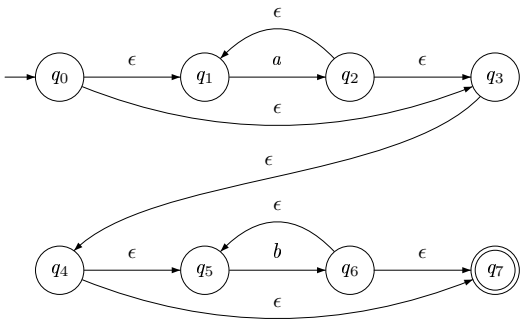

Figura 3.46. Autômato que reconhece a^*b^*

□

Um **grafo de expressões** é uma máquina de estados na qual as transições entre os estados são rotuladas por expressões regulares construídas sobre um mesmo alfabeto Σ. Formalmente:

$$N = (Q, \Sigma, \delta', q_0, F)$$

em que:

- Q, Σ, q_0 e F são definidos como no caso dos autômatos finitos;
- $\delta' : Q \times ER_\Sigma \rightarrow 2^Q$, onde ER_Σ representa o conjunto de todas as expressões regulares que podem ser definidas sobre o alfabeto Σ.

Note-se que todo autômato finito é, por definição, um caso particular de um grafo de expressões em que as expressões regulares entre os estados são reduzidas aos casos elementares σ, ϵ e $\varnothing$.

De forma similar ao caso do autômato finito, define-se a linguagem aceita por um grafo de expressões como sendo aquela que é gerada pela união de todas as expressões regulares que podem ser obtidas pela concatenação das expressões regulares que rotulam algum caminho entre o estado inicial e algum estado final do grafo.

A obtenção sistemática de uma expressão regular que gera linguagem aceita por um autômato finito pode ser feita através de um método que elimina, um a um, todos os estados do autômato que não sejam o inicial ou o final. Como decorrência da eliminação desses estados, as transições originais são substituídas por expressões regulares que preservam a linguagem aceita pelo dispositivo, e o correspondente autômato torna-se, finalmente, um grafo de expressões com apenas dois estados.

Teorema 3.12 (Expressões regulares $\Leftarrow$ autômatos finitos) *Seja M um autômato finito. Então existe uma expressão regular r que gera a linguagem aceita por M.*

Justificativa A expressão regular que gera a linguagem aceita por M pode ser obtida através da aplicação do Algoritmo 3.17, que especifica as regras de mapeamento de M em uma série de grafos de expressões com quantidades sucssivamente menores de estados. Ao se atingir um grafo com apenas dois estados, a expressão regular correspondente à linguagem aceita pelo autômato original pode ser inferida trivialmente.

O algoritmo apresentado a seguir permite a eliminação sistemática dos estados que não sejam finais ou inicial de um autômato qualquer (não-determinístico e com transições em vazio, no caso geral). À medida em que esses estados são eliminados, criam-se novas transições na forma de expressões regulares e o autômato torna-se um grafo de expressões.

A figura 3.47 representa a contextualização de um estado x a ser eliminado em um grafo de expressões. Ela ilustra todas as situações que podem ocorrer quando se consideram os estados precessores e os estados sucessores de x, assim como as transições entre x e esses estados.

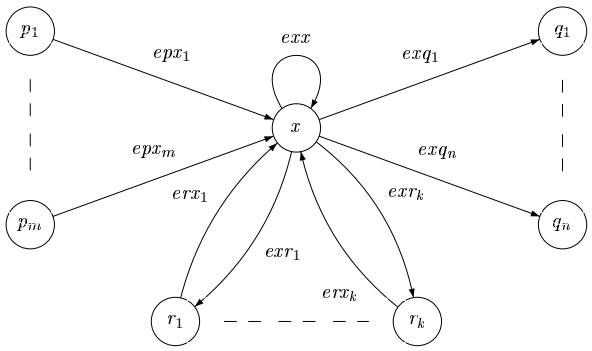

Figura 3.47. Contextualização de um x estado a ser eliminado

Considera-se, genericamente, que o estado x possui m estados que são apenas predecessores (identificados por p_1 até p_m), n estados que são apenas sucessores (q_1 até q_n) e k estados que são simultaneamente predecessores e sucessores. O estado x possui, portanto:

- $m + k$ estados precessores;
- $n + k$ estados sucessores.

Os rótulos das transições são representados por nomes que identificam o estado de origem e o estado de destino da mesma. Assim, por exemplo, a transição epx_1 tem como origem o estado p_1 e como destino o estado x, e a transição exq_n tem como origem o estado x e como destino o estado q_n. A transição exx tem como origem e destino o estado x.

As transições do grafo de expresões são inicialmente rotuladas da mesma forma que as transições do autômato finito correspondente, ou seja, com os símbolos do alfabeto Σ (para as transições não-vazias), e com com símbolo ϵ (para as transições vazias).

A eliminação de um estado x em um grafo de expressões (assim como das transições que tem o estado x como origem ou como destino) é feita de forma a garantir que a linguagem aceita pelo autômato original não sofra alterações.

Algoritmo 3.17 (Expressão regular $\Leftarrow$ autômato) *Obtenção de uma expressão regular a partir de um autômato finito.*

- Entrada: um autômato finito $M = (Q, \Sigma, \delta, q_0, F)$;
- Saída: uma expressão regular r tal que $r = L(M)$;

- Método:

  1. Se $\left|Q\right| > 1$, obter $M' = (Q', \Sigma, \delta', q_0, F')$ tal que $L(M') = L(M)$ e $\left|Q\right| = 1$:

     a) $Q' \leftarrow Q \cup \{q_F\}$;

     b) $\delta' \leftarrow \delta \cup \{(q_i, \epsilon) \rightarrow q_F) \mid q_i \in F\}$;

     c) $F' \leftarrow \{q_F\}$;

     Caso contrário, considerar $M' = M$.

  2. Para cada um dos estados $x \in (Q' - \{q_0, q_F\})$, fazer:

     a) Considerar que os estados predecessores, antecessores e respectivas transições associadas ao estado x estejam identificados conforme a figura 3.47.

     b) Para $1 \leq i \leq m$, fazer:

        i. Para $1 \leq j \leq n$, fazer $\delta' \leftarrow \delta' \cup \{(p_i, epx_i exx^* exq_j) \rightarrow q_j\}$;

        ii. Para $1 \leq j \leq k$, fazer $\delta' \leftarrow \delta' \cup \{(p_i, epx_i exx^* exr_j) \rightarrow r_j\}$;

     c) Para $1 \leq i \leq k$, fazer:

        i. Para $1 \leq j \leq n$, fazer $\delta' \leftarrow \delta' \cup \{(r_i, erx_i exx^* exq_j) \rightarrow q_j\}$;

        ii. Para $1 \leq j \leq k$, fazer $\delta' \leftarrow \delta' \cup \{(r_i, erx_i exx^* exr_j) \rightarrow r_j\}$;

     d) $\forall\, q_i, q_j \in Q', \sigma \in (\Sigma \cup \{\epsilon\})$:

        i. $\delta' \leftarrow \delta' - \{(q_i, x) \rightarrow q_j\}$;

        ii. $\delta' \leftarrow \delta' - \{(x, q_i) \rightarrow q_j\}$;

        iii. $Q' \leftarrow Q' - \{x\}$.

As novas transições geradas através desse processo são expressões regulares sobre o alfabeto Σ do autômato original, e englobam as operações de concatenação e fechamento reflexivo e transitivo. Caso existam duas ou mais transições entre um mesmo estado de origem um mesmo estado de destino, essas deverão ser substituídas pela expressão regular que representa a união das expressões originais, conforme indicado na figura 3.48.

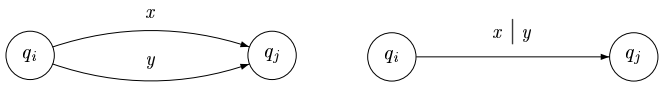

Figura 3.48. Combinação de transições que partem e chegam de estados idênticos

O processo de eliminação de estados prossegue até que todos os estados que não sejam finais nem inicial tenham sido eliminados. Nessas condições, o autômato resultante terá apenas dois estados e no máximo quatro transições, conforme a figura 3.49.

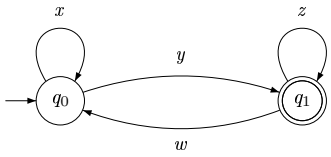

Figura 3.49. Autômato com dois estados que aceita a linguagem $x^*yz^*(wx^*yz^*)$

A inspeção da figura 3.49 permite inferir imediatamente a expressão regular que representa a linguagem aceita pelo grafo de expressões, e, conseqüentemente, a linguagem aceita pelo autômato finito original: $x^*yz^*(wx^*yz^*)$. ■

Exemplo 3.41 O autômato da Figura 3.50 reconhece a linguagem:

$$L = \{w \in \{a, b, c\}^* \mid w \text{ não contém a subcadeia } abc\}$$

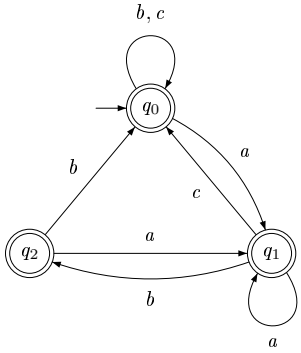

Figura 3.50. Autômato com múltiplos estados finais

A construção de uma expressão regular que gera a linguagem aceita por M inicia com a obtenção de um autômato equivalente com um único estado final, conforme o Algoritmo 3.17. O resultado é apresentado na Figura 3.51.

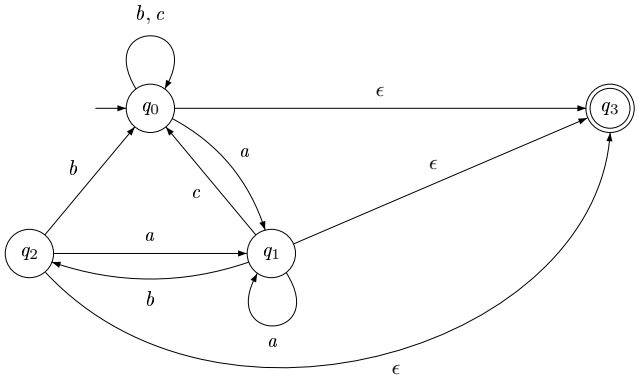

Figura 3.51. Autômato equivalente com um único estado final

A eliminação do estado q_2 resulta no grafo de expressões da Figura 3.52.

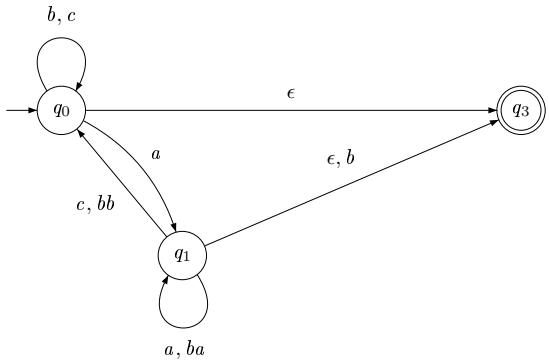

Figura 3.52. Grafo de expressões equivalente, após a eliminação do estado q_2

A eliminação do estado q_1 resulta no grafo de expressões da Figura 3.52.

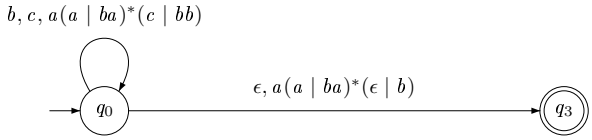

Figura 3.53. Grafo de expressões equivalente, após a eliminação dos estados q_2 e q_1

A análise da Figura 3.53 permite, finalmente, inferir a expressão regular que representa a linguagem aceita pelo autômato da Figura 3.50:

$$(\underbrace{b} \mid \underbrace{c} \mid \underbrace{a(a \mid ba)^*(c \mid bb)})^*(\underbrace{\epsilon} \mid \underbrace{a(a \mid ba)^*(\epsilon \mid b)})$$

As Figuras 3.54 e 3.55 representam, respectivamente, os grafos de expressão intermediários correspondentes caso a ordem de eliminação dos estados seja invertida (isto é, primeiro o estado q_1 e depois o estado q_2).

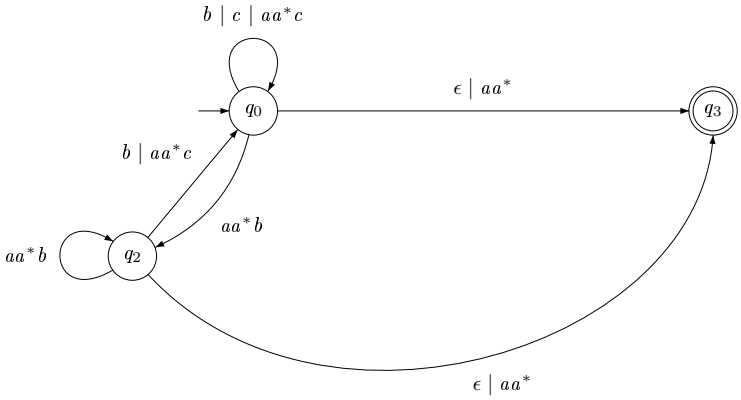

Figura 3.54. Grafo de expressões equivalente, após a eliminação do estado q_1

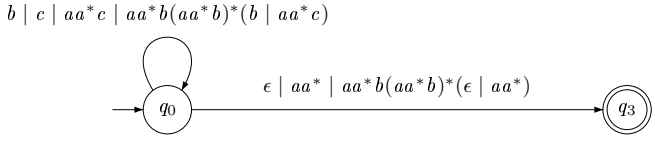

Figura 3.55. Grafo de expressões equivalente, após a eliminação dos estados q_1 e q_2

O resultado obtido é uma expressão regular equivalente porém diferente da anterior:

$$(\underbrace{b}_{} \mid \underbrace{c}_{} \mid \underbrace{aa^*c}_{} \mid \underbrace{aa^*b(aa^*b)^*(b \mid aa^*c)}_{})^*(\underbrace{\epsilon}_{} \mid \underbrace{aa^*}_{} \mid \underbrace{aa^*b(aa^*b)^*(\epsilon \mid aa^*)}_{})$$

□

Os resultados apresentados até este ponto são suficientes para estabelecer intuitivamente a equivalência completa dos conjuntos (expressões) regulares com as gramáticas lineares à direita e também com os autômatos finitos, no que se refere à classe de linguagens que tais dispositivos são capazes de representar.

A Tabela 3.41 apresenta um resumo que oferece uma visão abrangente das várias possibilidades de conversão direta entre os diversos formalismos estudados para representar as linguagens regulares, conforme discutido nas Seções 3.1 a 3.6.

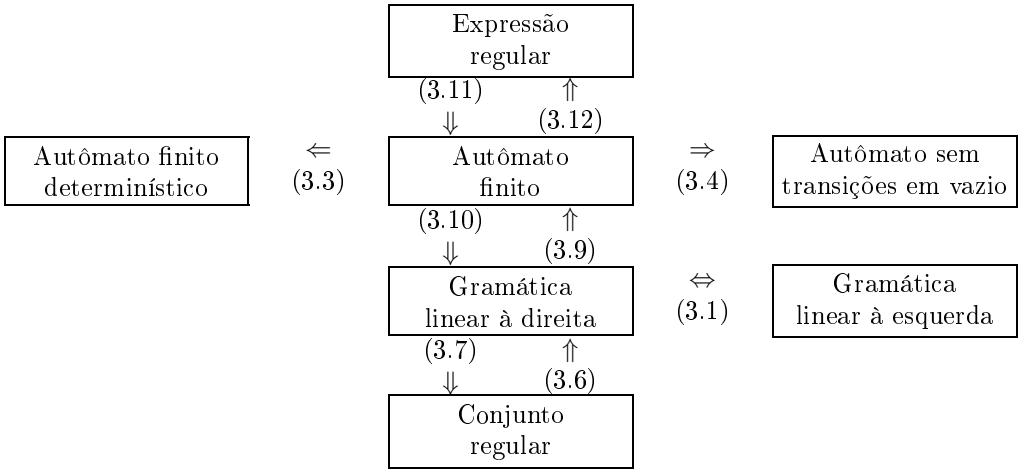

Tabela 3.41. Visão geral da equivalência dos formalismos e respectivos teoremas

3.7 Minimização de Autômatos Finitos

Um importante resultado da teoria dos autômatos refere-se à equivalência da classe dos autômatos finitos determinísticos com a dos não-determinísticos, ou seja, não-determinismos em autômatos finitos em nada contribuem para ampliar a classe de linguagens por eles reconhecíveis. Conforme mostrado anteriormente, é sempre possível transformar autômatos finitos não-determinísticos que contenham transições em vazio em outros equivalentes, determinísticos, isentos de tais transições.

Nesta seção é apresentado talvez o mais importante resultado teórico conhecido para a classe das linguagens regulares: o da existência de autômatos finitos determinísticos, mínimos e únicos, que reconhecem os respectivos conjuntos regulares. Em outras palavras, pode-se provar que cada conjunto regular é reconhecido por um autômato finito mínimo e único. O termo **mínimo** é empregado para designar um autômato finito que tenha o número mínimo possível de estados.

A importância desse resultado decorre de uma série de fatores. Em primeiro lugar, porque foi demonstrado que ele é válido apenas para a classe das linguagens definidas por autômatos finitos, não havendo correspondente para outras classes de linguagens; em segundo lugar, porque ele extrapola o interesse puramente teórico, despertando muito in-