Conjuntos e Expressões Regulares

O texto apresentado neste documento é uma adaptação (tradução) de:

Thomas A. Sudkamp. 1997. Languages and machines: an introduction to the theory of computer science. Addison-Wesley Longman Publishing Co., Inc., USA.

Vimos que definições recursivas fornecem uma ferramenta para definir as strings de uma linguagem. Mas não são ferramentas adequadas para impor os requisitos sintáticos de linguagens complexas, como linguagens matemáticas ou de computador.

Outra técnica para construir linguagens é usar operações para construir conjuntos complexos de strings a partir de outros mais simples. Uma operação definida em strings pode ser estendida a uma operação em conjuntos, portanto em linguagens. Descrições de linguagens infinitas podem então ser construídas a partir de conjuntos finitos usando as operações de conjunto.

1. Operações sobre Linguagens

Definição 1.1

A concatenação de linguagens X e Y, denotada como XY, é a linguagem

XY = { uv | u ∈ X e v ∈ Y }.

A concatenação de X consigo própria *n* vezes é denotada X^n. X^0 é definido como { λ }.

Exemplo 1.1

Seja X = *{a, b, c}* e Y = *{abb, ba}*. Então

XY = *{aabb, babb, cabb, aba, bba, cba}*

X^0 = { λ }

X^1 = X = *{a, b, c}*

X^2 --= XX = {*aa, ab, ac, ba, bb, bc, ca, cb, cc*}

X^3 = X^2X = *{aaa, aab, aac, aba, abb, abc, aca, acb, acc, baa, bab, bac, bba, bbb, bbc, bca, bcb, bcc, caa, cab, cac, cba, cbb, cbc, cca, ccb, ccc}.* ❑

Os conjuntos no exemplo anterior devem parecer familiares. Para cada i, X^i contém as cadeias de comprimento i em Σ*. Essa observação leva a outra operação de conjunto, denominada de Fecho de Kleene de um conjunto X, e denotada por X*. Usando o operador *, as strings sobre um conjunto podem ser definidas com as operações de concatenação e união.

Definição 1.1

Seja X um conjunto. Então:

$$X^* = \bigcup_{i=0}^{\infty} X^i \qquad \text{e} \qquad X^+ = \bigcup_{i=1}^{\infty} X^i.$$

X* contém todas as strings que podem ser construídas a partir dos elementos de X.

X^+ é o conjunto de strings não nulas construídas a partir de X.

Uma definição alternativa de X^+ usando concatenação e o fecho de Kleene é X^+ = XX*.

A definição de linguagens requer a especificação não ambígua das strings que pertencem à linguagem. A descrição de linguagens de maneira informal carece do rigor necessário para uma definição precisa. Por exemplo, considere a linguagem sobre {a, b} consistindo em todas as strings que contêm a substring bb. Isso significa que uma string na linguagem contém exatamente uma ocorrência de bb ou várias substrings bb são permitidas? Isso pode ser respondido descrevendo especificamente as strings como contendo exatamente uma ou pelo menos uma ocorrência de bb. No entanto, esses tipos de perguntas são inerentes ao meio impreciso fornecido pelas línguas naturais.

A precisão proporcionada por operações de conjunto pode ser usada para fornecer uma descrição inequívoca das strings de uma linguagem. O resultado de uma operação unária em uma linguagem ou de uma operação binária em duas linguagens define outra linguagem. O Exemplo 1.2 fornece uma definição teórica de conjunto das strings que contêm a substring bb. Nesta definição, está claro que a linguagem contém todas as strings nas quais bb ocorre pelo menos uma vez.

Exemplo 1.2

A linguagem

$$L = \{a, b\}^*\{bb\}\{a, b\}^*$$

consiste nas strings sobre {a, b} que contêm a substring bb. A concatenação do conjunto {bb} garante a presença de bb em todas as strings em L. Os conjuntos {a, b}* permitem que qualquer número de a's e b's, em qualquer ordem, preceda e siga a ocorrência de bb. ❑

Exemplo 1.3

O conjunto {aa, bb, ab, ba}* consiste em todas as strings de comprimento par sobre {a, b}. A concatenação repetida constrói strings adicionando dois elementos de cada vez.

O conjunto de strings de comprimento ímpar pode ser definido concatenando um único símbolo às strings de comprimento par: {a, b} {aa, bb, ab, ba}*. ❑

2. Conjuntos Regulares e Expressões Regulares

Na seção anterior, usamos operações de conjunto para construir novas linguagens a partir de outras já existentes. Os operadores foram selecionados para garantir que certos padrões ocorressem nas sequências da linguagem. Nesta seção, seguimos a abordagem de construção de linguagens a partir de operações de conjunto, mas limitamos os conjuntos e operações que são permitidos no processo de construção.

Vamos introduzir o conceito de *conjunto regular*. Um conjunto é regular se pode ser gerado a partir do conjunto vazio, o conjunto que contém a string nula e os elementos do alfabeto usando união, concatenação e a operação fecho de Kleene. Os conjuntos regulares são uma importante família de linguagens, ocorrendo tanto na teoria da linguagem formal quanto na teoria das máquinas de estado finito.

Definição 2.1

Seja Σ um alfabeto. Os conjuntos regulares sobre Σ são definidos recursivamente da seguinte forma:

i) Base: 0, { λ } e {a}, para todo a ∈ Σ, são conjuntos regulares sobre Σ.

ii) Etapa recursiva: sejam X e Y conjuntos regulares sobre Σ. Os conjuntos

X U Y

XY

X*

são conjuntos regulares sobre Σ.

iii) Fechamento: X é um conjunto regular sobre Σ apenas se puder ser obtido dos elementos da base por um número finito de aplicações do passo recursivo.

Os conjuntos definidos nos exemplos 1.2 e 1.3 são conjuntos regulares, por definição.

Exemplo 2.1

O conjunto de strings que começam e terminam com um *a* e contêm pelo menos um *b* é regular em {*a,b*}. As strings nesse conjunto podem ser descritas intuitivamente como "um *a*, seguido por qualquer string, seguido por um *b*, seguido por qualquer string, seguido por um *a*". A concatenação a seguir exibe a regularidade do conjunto:

$$\{a\}\{a,b\}^*\{b\}\{a,b\}^*\{a\}$$

❑

Por definição, conjuntos regulares são aqueles que podem ser construídos a partir do conjunto vazio, o conjunto que contém a string nula e os conjuntos que contêm um único elemento do alfabeto usando as operações de união, concatenação e fecho de Kleene.

Expressões regulares são usadas para abreviar as descrições de conjuntos regulares. O conjunto regular {*b*} é representado por ***b***, eliminando a necessidade dos colchetes {}. As operações de conjunto de união, fecho de Kleene e concatenação são designadas por U, * e justaposição, respectivamente. Os parênteses são usados para indicar a ordem das operações.

Definição 2.2

Seja Σ um alfabeto. As expressões regulares em Σ são definidas recursivamente da seguinte forma:

i) Base: ∅, λ e *a*, para cada *a* ∈ Σ, são expressões regulares sobre Σ.

ii) Etapa recursiva: Sejam *u* e *v* expressões regulares sobre Σ. As expressões

(u U v)

(uv)

(u*)

são expressões regulares sobre Σ.

iii) Fechamento: *u* é uma expressão regular sobre Σ somente se puder ser obtida a partir dos elementos da base por um número finito de aplicações do passo recursivo.

Como a união e a concatenação são associativas, os parênteses podem ser omitidos nas expressões que consistem em uma sequência de uma dessas operações. Para reduzir ainda mais o número de parênteses, uma precedência é atribuída aos operadores. A operação fecho de Kleene é arbitrada como a de maior prioridade, seguida pela concatenação e união. Empregando essas convenções, as expressões regulares para os conjuntos nos Exemplos 1.2, 1.3 e 2.1 são:

Conjunto Regular	Expressão Regular
{a, b}* {bb} {a, b}*	(a U b)* bb (a U b)*
{a, b} {aa, bb, ab, ba}*	(a, b) (aa, bb, ab, ba)*
{a} {a, b}* {b} {a, b}* {a}	a (a U b)* b (a U b)* a

A notação u^+ é usada para abreviar a expressão uu*. Da mesma forma, u^2 denota a expressão regular uu, u^3 denota u^2u, etc.

Outros exemplos de expressões regulares:

Expressão Regular	Especificação em português
(a U b U c)* aa (a U b U c)*	Strings sobre o alfabeto {a,b,c} que contêm 2 a's seguidos.
(a U b U c)* a (a U b U c)* a (a U b U c)*	Strings sobre o alfabeto {a,b,c} que contêm pelo menos 2 a's.
(b U c)* a (b U c)* a (a U b U c)*	Mesma linguagem acima.
(a U b U c)* a (b U c)* a (b U c)*	Mesma linguagem acima.
(b U c)* a (a U b U c)* a (b U c)*	Mesma linguagem acima.
(b U c)* a (b U c)* a (b U c)*	Strings sobre o alfabeto {a,b,c} que contêm exatamente 2 a's.
(aa U ab U ba U bb)*	Strings sobre o alfabeto {a,b} com comprimento par.
((a U b) (a U b))*	Mesma linguagem acima.
a* (b a* b a*)*	Strings sobre o alfabeto {a,b} com número par de b's.

Exemplo 2.2

Uma expressão regular é construída para representar o conjunto de strings sobre {a, b} que não contêm a substring aa. Uma string nesse conjunto pode conter um prefixo de qualquer número de b's. Todos os a's devem ser seguidos por pelo menos um b ou encerrar a string. A expressão

$$b^* (a b^+)^* \cup b^* (ab^+)^* a$$

gera o conjunto desejado, particionando-o em dois subconjuntos disjuntos; o primeiro consiste em strings que terminam em b e o segundo em strings que terminam em a. ❑

A tabela ao lado apresenta identidades observadas em expressões regulares, que podem ser usadas para simplificar uma expressão ou apresentar uma especificação alternativa equivalente.

Abaixo, essas identidades são aplicadas na expressão apresentada no Exemplo 2.2.

$$\begin{aligned} & b^*(ab^+)^* \cup b^*(ab^+)^*a \\ &= b^*(ab^+)^*(\lambda \cup a) \\ &= b^*(abb^*)^*(\lambda \cup a) \\ &= (b \cup ab)^*(\lambda \cup a). \end{aligned}$$

1. $\emptyset u = u\emptyset = \emptyset$
2. $\lambda u = u\lambda = u$
3. $\emptyset^* = \lambda$
4. $\lambda^* = \lambda$
5. $u \cup v = v \cup u$
6. $u \cup \emptyset = u$
7. $u \cup u = u$
8. $u^* = (u^*)^*$
9. $u(v \cup w) = uv \cup uw$
10. $(u \cup v)w = uw \cup vw$
11. $(uv)^*u = u(vu)^*$
12. $(u \cup v)^* = (u^* \cup v)^* = u^*(u \cup v)^* = (u \cup vu^*)^* = (u^*v^*)^* = u^*(vu^*)^* = (u^*v)^*u^*$