Azzolini Riccardo 2020-11-05

Espressioni regolari

1 Operazioni sui linguaggi

Prima di introdurre le espressioni regolari (un formalismo per definire linguaggi regolari), è necessario definire alcune operazioni sui linguaggi, che serviranno per dare poi il significato a tali espressioni.

1.1 Unione

Dati due linguaggi L_1, L_2, la loro **unione** $L_1 \cup L_2$ è l'unione insiemistica dei due linguaggi:

$$L_1 \cup L_2 = \{w \mid w \in L_1 \text{ o } w \in L_2\}$$

I linguaggi L_1 e L_2 possono essere definiti su alfabeti diversi: $L_1 \subseteq \Sigma_1^*$ e $L_2 \subseteq \Sigma_2^*$. In tal caso, l'unione dei linguaggi è definita sull'unione insiemistica dei due alfabeti: $L_1 \cup L_2 \subseteq (\Sigma_1 \cup \Sigma_2)^*$. Ad esempio, dati i linguaggi $L_1 = \{0\}^*$ e $L_2 = \{1\}^*$, la loro unione è:

$$L_1 \cup L_2 = \{w \in \{0,1\}^* \mid w = 0 \ldots 0 \text{ o } w = 1 \ldots 1\}$$

1.2 Concatenazione

Dati due linguaggi L_1, L_2, la loro **concatenazione**, indicata con $L_1 \cdot L_2$ o semplicemente $L_1 L_2$, è il linguaggio che contiene tutte le stringhe ottenute concatenando una stringa di L_1 con una stringa di L_2:

$$L_1 L_2 = \{wv \mid w \in L_1 \text{ e } v \in L_2\}$$

Ad esempio, se $L_1 = \{001, 10, 11\}$ e $L_2 = \{\epsilon, 001\}$, la concatenazione di questi due linguaggi è:

$$\begin{aligned} L_1 L_2 &= \{001\epsilon, 10\epsilon, 11\epsilon, 001001, 10001, 11001\} \\ &= \{001, 10, 11, 001001, 10001, 11001\} \end{aligned}$$

Considerando invece $L_1 = \{0\}^*$ e $L_2 = \{1\}^*$, si ha che:

$$L_1 L_2 = \{w \in \{0,1\}^* \mid w = \underbrace{0 \ldots 0}_{n} \underbrace{1 \ldots 1}_{m},\ n \geq 0,\ m \geq 0\}$$

1.3 Elevamento a potenza

Dato un linguaggio L, la **potenza** i-esima di L (dove i è un numero intero maggiore o uguale a 0), indicata con L^i, è definita induttivamente su i:

$$L^i = \begin{cases} \{\epsilon\} & \text{se } i = 0 \\ L^{i-1}L & \text{se } i > 0 \end{cases}$$

Considerando ad esempio $L = \{0, 11\}$, alcune sue potenze sono:

$$\begin{aligned} L^0 &= \{\epsilon\} \\ L^1 &= L^0 L = \{\epsilon\}\{0, 11\} = \{0, 11\} \\ L^2 &= L^1 L = \{0, 11\}\{0, 11\} = \{00, 011, 110, 1111\} \\ L^3 &= L^2 L = \{00, 011, 110, 1111\}\{0, 11\} \\ &= \{000, 0011, 0110, 01111, 1100, 11011, 1110, 11111\} \end{aligned}$$

Come si può vedere da questo esempio, si ha in generale che $L^1 = L$.

Un caso particolare di elevamento a potenza è quello in cui si parte dal linguaggio che è l'insieme vuoto, $L = \varnothing$:

$$\begin{aligned} L^0 &= \{\epsilon\} \\ L^1 &= L^0 L = \{\epsilon\}\varnothing = \{wv \mid w = \epsilon \text{ e } \underbrace{v \in \varnothing}_{\text{non esiste}}\} = \varnothing \\ L^2 &= L^1 L = \varnothing\varnothing = \varnothing \\ &\vdots \\ L^n &= \varnothing \end{aligned}$$

Si osserva che la potenza L^0 *non* è l'insieme vuoto: è un insieme (linguaggio) che invece contiene una stringa, la stringa vuota ϵ. Invece, tutte le potenze successive sono l'insieme vuoto.

1.4 Chisure di Kleene

- La **chiusura di Kleene** di un linguaggio L, indicata con L^*, è definita come

  $$L^* = \bigcup_{i \geq 0} L^i$$

  ovvero è l'unione di tutte le potenze di L.

  Siccome $L^0 = \{\epsilon\} \subseteq L^*$, la chiusura di Kleene di un qualunque linguaggio contiene sempre la stringa vuota ($\epsilon \in L^*$).

- La **chiusura positiva** (**di Kleene**) di L, indicata invece con L^+, è

$$L^+ = \bigcup_{i \geq 1} L^i$$

cioè l'unione di tutte le potenze di L tranne L^0.

L'unica differenza rispetto alla chiusura di Kleene è che L^+ contiene la stringa vuota ($\epsilon \in L^+$) solo se questa è già presente nel linguaggio di partenza ($\epsilon \in L$). Vale dunque la relazione $L^* = L^+ \cup \{\epsilon\}$.

Ad esempio, dati l'alfabeto $\Sigma = \{0, 1\}$ e il linguaggio che coincide con tale alfabeto, $L = \Sigma = \{0, 1\}$ (il linguaggio di tutte le possibili stringhe formate da un solo simbolo dell'alfabeto), la potenza i-esima di L è l'insieme di tutte le stringhe di lunghezza i sull'alfabeto, quindi:

- La chiusura di Kleene di L è l'insieme di tutte le possibili stringhe sull'alfabeto:

$$\begin{aligned} L^* &= L^0 \cup L^1 \cup L^2 \cup L^3 \cdots \\ &= \{\epsilon\} \cup \{0, 1\} \cup \{00, 01, 10, 11\} \cup \{000, 001, 010, 011, 100, 101, 110, 111\} \cup \cdots \\ &= \{0, 1\}^* \end{aligned}$$

Questo giustifica formalmente la notazione Σ^*, che era stata inizialmente introdotta in modo intuitivo.

- La chiusura positiva di L è

$$\begin{aligned} L^+ &= L^1 \cup L^2 \cup L^3 \cdots \\ &= \{0, 1\} \cup \{00, 01, 10, 11\} \cup \{000, 001, 010, 011, 100, 101, 110, 111\} \cup \cdots \end{aligned}$$

cioè, in questo caso, $L^+ = L^* \setminus \{\epsilon\}$.

Un altro esempio interessante è il caso di $L = \varnothing$:

$$\begin{aligned} L^* &= \{\epsilon\} \cup \varnothing \cup \varnothing \cup \cdots = \{\epsilon\} \\ L^+ &= \varnothing \cup \varnothing \cup \cdots = \varnothing \end{aligned}$$

In particolare, la chiusura di Kleene dell'insieme vuoto dà un linguaggio non vuoto, contenente la sola stringa vuota.

2 Espressioni regolari

Per capire più facilmente come sono definite le espressioni regolari, è utile fare un'analogia con le espressioni aritmetiche, come ad esempio

$$10 \cdot (3 + 5)$$

Queste ultime, infatti, sono formate da:

- degli elementi di base, le costanti che rappresentano i numeri (ad esempio 10, 3, 5);
- degli operatori (ad esempio $\cdot$ e $+$), che rappresentano delle operazioni sui numeri;
- le parentesi, che servono per variare la precedenza di applicazione degli operatori.

Nelle espressioni regolari si hanno elementi analoghi:

- delle costanti, che rappresentano dei linguaggi;
- degli operatori, che rappresentano operazioni sui linguaggi;
- le parentesi, usate come nelle espressioni aritmetiche.

Un'espressione aritmetica viene poi valutata in un numero; allo stesso modo, la valutazione di un'espressione regolare dà come risultato un linguaggio.

2.1 Definizione formale

Le **espressioni regolari** E su un alfabeto Σ, e il relativo *linguaggio generato*[1] $L(E)$, sono definiti induttivamente nel modo seguente:

2.1.1 Casi base

	Espressione	Linguaggio generato
	ϵ	$L(\epsilon) = \{\epsilon\}$
	$\varnothing$	$L(\varnothing) = \varnothing$
$\forall a \in \Sigma$	a	$L(a) = \{a\}$

È importante osservare che ϵ, ad esempio, nel contesto delle espressioni regolari è un simbolo che può comparire in tali espressioni, mentre nell'ambito dei linguaggi rappresenta una particolare stringa (la stringa vuota, senza simboli). Lo stesso vale per $\varnothing$ e a, che nelle espressioni regolari hanno la funzione di simboli, mentre nell'ambito dei linguaggi rappresentano rispettivamente il linguaggio vuoto e un simbolo dell'alfabeto Σ.

[1] Si dice che le espressioni regolari descrivono i linguaggi in modo generativo perché esse partono dalle componenti di base dei linguaggi, costruendo poi linguaggi man mano più "grandi" mediante l'applicazione di vari operatori.

2.1.2 Casi induttivi

Date due espressioni regolari R e S, si definiscono i seguenti operatori:

	Espressione	Linguaggio generato
unione	$R + S$	$L(R + S) = L(R) \cup L(S)$
concatenazione	RS	$L(RS) = L(R) \cdot L(S)$
star (di Kleene)	R^*	$L(R^*) = (L(R))^*$
parentesi	(R)	$L((R)) = L(R)$

Note:

- Le parentesi sono definite come un operatore semplicemente per comodità, e non cambiano il significato dell'espressione regolare che racchiudono.
- Quando servirà esplicitare l'operatore di concatenazione, lo si indicherà con il simbolo $\cdot$ (come la concatenazione di linguaggi).
- L'operatore * ha la precedenza più alta, seguito da $\cdot$ e infine $+$. Quando si hanno più operatori con la stessa precedenza, li si applica da sinistra a destra: ad esempio, l'espressione $a + b + c$ va letta come $(a + b) + c$. Per scrivere espressioni in cui gli operatori si applicano in ordini diversi, si usano le parentesi.

2.2 Esempi

Alcuni esempi di espressioni regolari su $\Sigma = \{0, 1\}$, con i relativi linguaggi generati, sono:

- $E = 0 + 1$

$$\begin{aligned} L(0 + 1) &= L(0) \cup L(1) \\ &= \{0\} \cup \{1\} \\ &= \{0, 1\} \end{aligned}$$

- $E = (0 + 1)^*00$

$$\begin{aligned} L((0 + 1)^*00) &= L((0 + 1)^*0) \cdot L(0) \\ &= L((0 + 1)^*) \cdot L(0) \cdot L(0) \\ &= L(0 + 1)^* \cdot L(0) \cdot L(0) \\ &= \{0, 1\}^* \cdot \{0\} \cdot \{0\} \\ &= \{0, 1\}^* \cdot \{00\} \\ &= \{w00 \mid w \in \{0, 1\}^*\} \end{aligned}$$

3 Dai DFA alle espressioni regolari

Una volta definite le espressioni regolari, bisogna determinare quale sia la classe di linguaggi che esse possono generare. Si dimostrerà che tale classe corrisponde alla classe dei linguaggi regolari, cioè quelli che possono essere riconosciuti da un DFA. Il primo verso di questa corrispondenza è dato dal seguente teorema:

Teorema: Se un linguaggio L è riconosciuto da un DFA A, $L = L(A)$, allora esiste un'espressione regolare R che genera lo stesso linguaggio, cioè tale che $L = L(R)$.

La dimostrazione di questo teorema non verrà data (poiché non è particolarmente importante per le applicazioni), ma è importante sapere che si tratta di una dimostrazione costruttiva: esiste effettivamente un metodo algoritmico per costruire, a partire da un DFA A, l'espressione regolare R tale che $L(R) = L(A)$.