§1 词法分析

1.1. 语言. 语言是字符串的集合.

定义 1.1 (字母表)**.** 字母表 Σ 是一个有限的符号集合.

现在, 符号没有任何的含义. 符号是什么意思是语义干的事情.

定义 1.2 (字母表)**.** 字母表 Σ 上的串 (s) 是由 Σ 中符号构成的一个有穷序列.

其中特殊的串是空串, $|\epsilon| = 0$

定义 1.3 (串上的连接运算)**.** 例如 $x = \text{dog}, y = \text{house} \quad xy = \text{doghouse}$, $s\epsilon = \epsilon S = s$

定义 1.4 (串上的指数运算)**.**

$$s^0 \triangleq \epsilon$$
$$s^i \triangleq ss^{i-1}, i > 0$$

定义 1.5. 语言是给定字母表 Σ 上一个任意的可数的串集合。

注:

1. 注意 $\emptyset$ 和 $\{\epsilon\}$. 后者是只有空串的语言.
2. 例如 ws : {blank, tab, newline }
3. 这就可以通过集合操作构造新的语言.

定义 1.6. 我们可以通过如下的方式构造新语言.

运算	定义和表示
L 和M 的并	$L \cup M := \{s \mid s$ 属于L 或者s 属于$M\}$
L 和M 的连接	$LM := \{st \mid s$ 属于L 且t 属于$M\}$
L 的 Kleene 闭包	$L^* := \cup_{i=0}^{\infty} L^i$
L 的正闭包	$L^+ := \cup_{i=1}^{\infty} L^i$

注: L^* (L^+) 允许我们构造无穷集合

例子 1.1. 假设 $L = \{A, B, \ldots, Z, a, b, \ldots, z\}, D = \{0, 1, \ldots, 9\}$. 那么

- $L \cup D =$ 所有的大小写字母和所有的一位数码构成的集合.
- $|LD| = 52 \times 10 = 520$.
- $L^4 =$ 所有长度为 4 的由字母构成的语言的集合.
- $L^*=$ 所有字母构成的字符串 (包含空串) 构成的集合.
- $D^+ =$ 所有数字构成的字符串 (不包含空串) 构成的集合.
- $L(L \cup D)^* =$ 以字母开头, 跟上 0 个或者若干个字母或者数字的字符串构成的集合. (也就是类似于 id).

1.2. 正则表达式. 注意区分语法和语义. 例如正则表达式中, 每个正则表达式 r 对应一个正则语言 $L(r)$. 正则表达式是语法, 正则语言是语义.

定义 1.7 (正则表达式)**.** 给定字母表 Σ, Σ 上的正则表达式由且仅由以下规则定义:

1. ϵ 是正则表达式;
2. $\forall a \in \Sigma, a$ 是正则表达式;
3. 如果 r 是正则表达式, 则 (r) 是正则表达式;
4. 如果 r 与 s 是正则表达式, 则 $r \mid s, rs, r^*$ 也是正则表达式.

运算优先级: ()$\succ$ $*\succ$ 连接 $\succ|$

例如

$$(a)\,|((b)^*(c)) \equiv a|\,b^*c.$$

每个正则表达式 r 对应一个正则语言 $L(r)$, 这代表了它的语义.

定义 1.8 (正则表达式对应的正则语言)**.**

$$\begin{aligned} L(\epsilon) &= \{\epsilon\} \\ L(a) &= \{a\}, \forall a \in \Sigma \\ L((r)) &= L(r) \\ L(r \mid s) = L(r) \cup L(s) \quad L(rs) &= L(r)L(s) \quad L\left(r^*\right) = (L(r))^* \end{aligned}$$

例子 1.2. 如果 $\Sigma = \{a, b\}$, $L(a \mid b) = \{a, b\}$. 那么

- $L\left(a^*\right) = \{\epsilon, a, aa, aaa, \cdots, \}$.
- $L\left((a \mid b)^*\right)$= 由 a 和 b 构成的任意长度的字符串 (包括空串).
- $L\left(a \mid a^*b\right) = \{a, b, ab, aab, aaab, \cdots\}$.

实际生活中, 我们的正则表达式的列表如下:

表达式	匹配	例子
c	单个非运算符字符 c	a
$\backslash c$	字符 c 的字面值	$\backslash *$
$"s$	串 s 的字面值	$"**"$
.	除换行符以外的任何字符 (看环境, ANTLR4 中可以匹配换行符)	$a.*b$
^	一行的开始	$\hat{}abc$
\$	行的结尾	abc\$
$[s]$	字符串 s 中的任何一个字符	[abc]
$[\hat{}s]$	不在串 s 中的任何一个字符	$\hat{}[abc]$
r^*	和 r 匹配的零个或多个串连接成的串	$a*$
$r+$	和 r 匹配的一个或多个串连接成的串	$a+$
$r?$	零个或一个 r	a ?
$r\{m, n\}$	最少 m 个, 最多 n 个 r 的重复出现	$a\{1, 5\}$
r_1r_2	r_1 后加上 r_2	ab
$r_1 \mid r_2$	r_1 或 r_2	$a \mid b$
(r)	与 r 相同	$(a \mid b)$
r_1/r_2	后面跟有 r_2 时的 r_1	$abc/123$

有一些简单记录方法 (在 Vim 中)

- $\backslash w$ 表示所有大小写字母, 数字, 以及下划线
- $\backslash W$ 表示除去所有大小写字母, 数字, 以及下划线
- $\backslash d$ 表示所有数码
- $\backslash D$ 表示除去所有数码

例子 1.3.

$$\left(0 \mid \left(1\,(01^*0)^*\,1\right)\right)^*$$

表示的二进制的 3 的倍数. 可以从`regex.com`中观察一些例子.

接下来介绍自动机.

定义 1.9 (NFA)**.** 非确定性有穷自动机 $\mathcal{A}$ 是一个五元组 $\mathcal{A} = (\Sigma, S, s_0, \delta, F)$:

1. 字母表 $\Sigma(\epsilon \notin \Sigma)$;
2. 有穷的状态集合 S;
3. (唯一) 的初始状态 $s_0 \in S$;
4. 状态转移函数 δ.
$$\delta : S \times (\Sigma \cup \{\epsilon\}) \to 2^S$$
5. 接受状态集合 $F \subseteq S$

注:

1. 非确定性指的是其 "出路" 可能不唯一. 此外, 可以接受空串然后进行转移 (称为 ϵ 转移).
2. 所有没有对应出边的字符默认指向"空状态"$\emptyset$. 此状态无论接受什么字符都到自身. 一旦进入将无法出去.

下面来看 NFA 的语义. (非确定性) 有穷自动机是一类极其简单的计算装置, 它可以识别 (接受/拒绝) Σ 上的字符串.

定义 1.10 (接受 (Accept))**.** (非确定性) 有穷自动机 $\mathcal{A}$ 接受字符串 x, 当且仅当存在一条从开始状态 s_0 到某个接受状态 $f \in F$ 、标号为 x 的路径.

因此, $\mathcal{A}$ 定义了一种语言 $L(\mathcal{A})$: 它能接受的所有字符串构成的集合.

关于自动机 $\mathcal{A}$ 的两个基本问题:

1. Membership 问题: 给定字符串 $x, x \in L(\mathcal{A})$?
2. $L(\mathcal{A})$ 究竟是什么?

定义 1.11 (DFA)**.** 确定性有穷自动机 $\mathcal{A}$ 是一个五元组 $\mathcal{A} = (\Sigma, S, s_0, \delta, F)$:

- 字母表 $\Sigma(\epsilon \notin \Sigma)$
- 有穷的状态集合 S
- 唯一的初始状态 $s_0 \in S$
- 状态转移函数 δ
$$\delta : S \times \Sigma \to S$$
- 接受状态集合 $F \subseteq S$

上述约定 (所有没有对应出边的字符默认指向一个"死状态") 同样适用于这里.

NFA 简洁易于理解, 便于描述语言 $L(\mathcal{A})$ DFA 易于判断 $x \in L(\mathcal{A})$, 适合产生词法分析器. 下面我们来走 RE $\Longrightarrow$ NFA $\Longrightarrow$ DFA $\Longrightarrow$ 词法分析器的流程.

a) 正则表达式到 NFA 要求 $L(N(r)) = L(r)$. 可以使用 Tompson 构造法. 使用四种基础的归纳.

$N(r)$ 的性质以及 Thompson 构造法复杂度分析

1. $N(r)$ 的开始状态与接受状态均唯一。
2. 开始状态没有人边, 接受状态没有出边。
3. $N(r)$ 的状态数 $|S| \leq 2 \times |r|$ 。
4. 每个状态最多有两个 ϵ - 入边与两个 ϵ-出边。
5. $\forall a \in \Sigma$, 每个状态最多有一个 a-人边与一个 a-出边。($|r|$: r 中运算符与运算分量的总和)