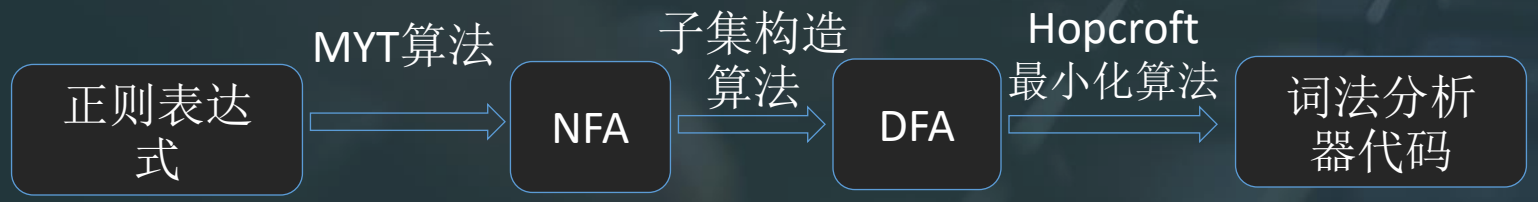
从正则表达式到DFA

2.2.1 基于MYT算法从正则表达式到NFA



2.2.1 基于MYT算法从正则表达式到NFA

MYT算法 (Thompson算法)

- McNaughton-Yamada-Thompson算法
- Thompson算法

基本思想: 对正则表达式的结构做归纳

- 对基本正则表达式直接构造NFA
- 对复合正则表达式递归构造NFA

2.2.1 基于MYT算法从正则表达式到NFA

给定字母表 ={c1, c2, …, cn}:

正则式	正则集/正则语言	备注
ε	{ε}	
c	{c}	$c \in \sum$

假设r和s是正则式，则以下也是正则式:

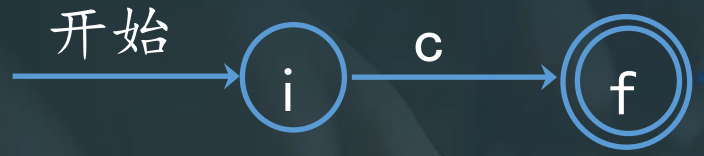
r | s	L(r)∪L(s)	或运算 (选择运算)
rs	L(r)L(s)	连接运算
r^*	$L(r)^*$	闭包运算
(r)	L(r)	括号运算

2.2.1 基于MYT算法从正则表达式到NFA

构造识别　的NFA

识别正则式 ε 的NFA

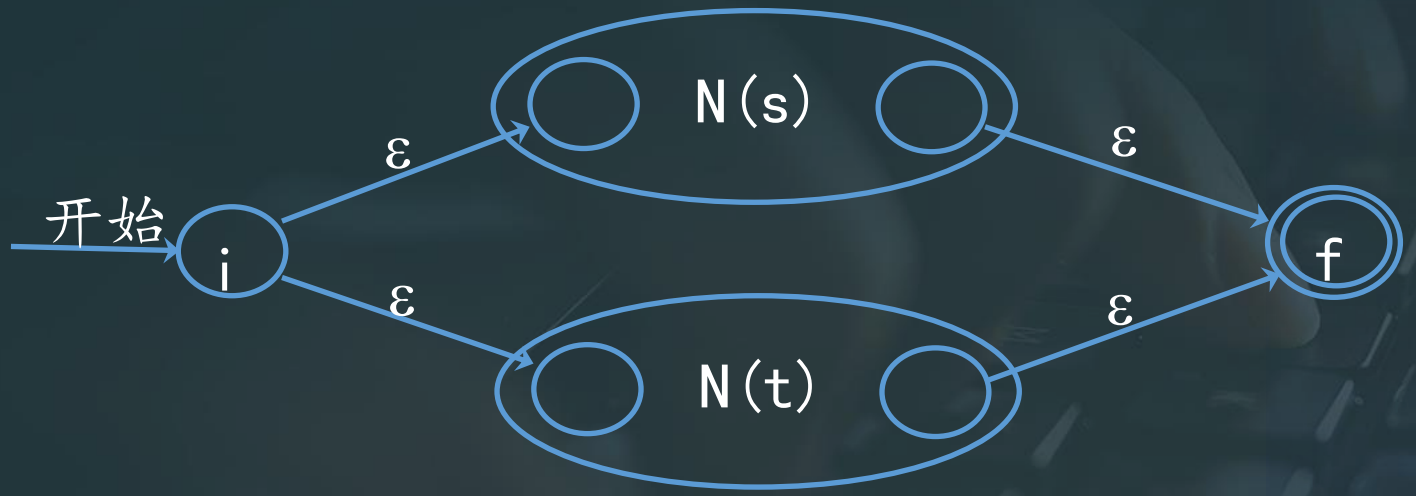
构造识别c的NFA



识别正则式 c 的NFA

- 特点：仅一个接受状态，它没有向外的转换

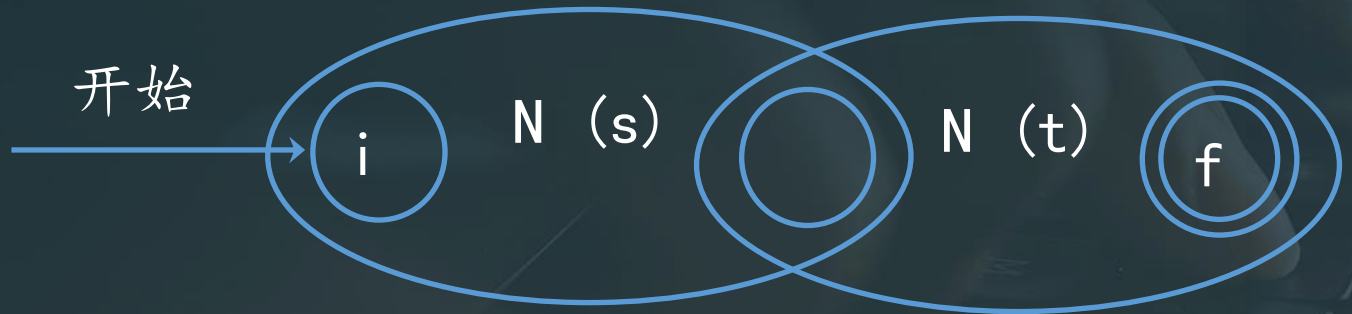
构造识别选择正则式 s|t 的NFA



识别正则式 s|t 的NFA

- 特点：仅一个接受状态，它没有向外的转换

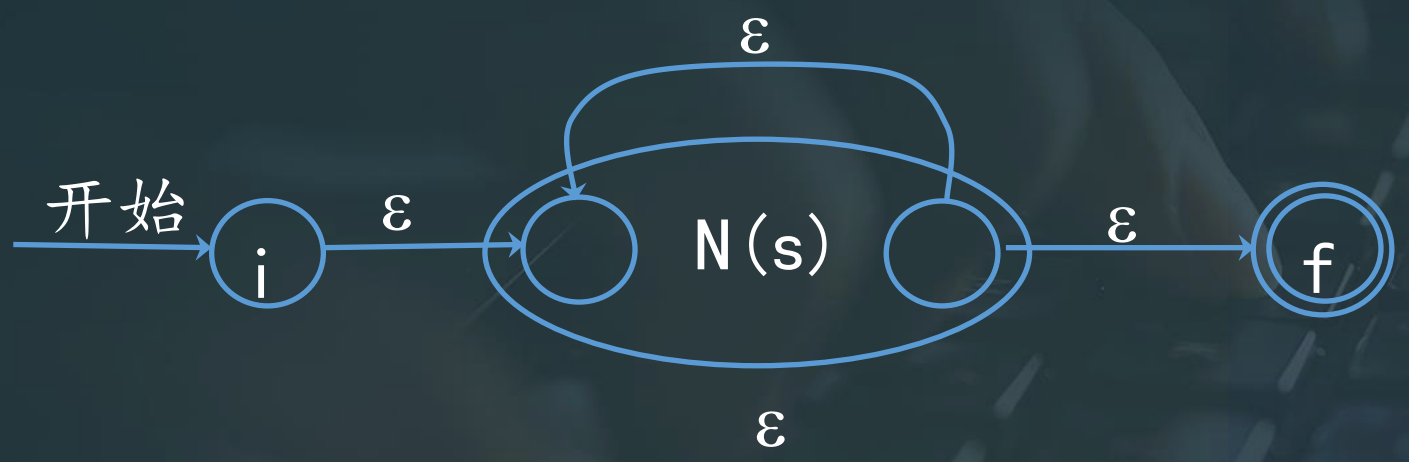
构造识别连接正则式 st 的NFA



识别正则式 st 的NFA

- 特点：仅一个接受状态，它没有向外的转换

构造识别闭包正则式 s^* 的NFA

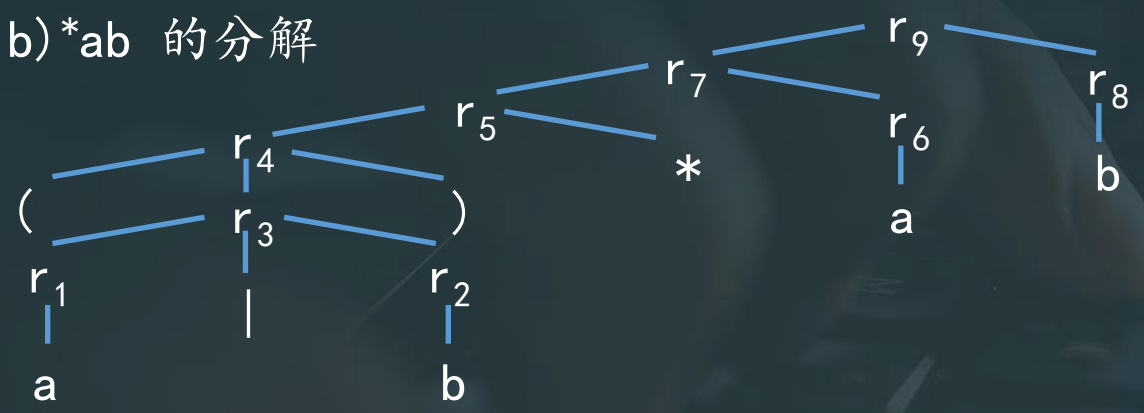


识别正则式 s^* 的NFA

- 特点：仅一个接受状态，它没有向外的转换

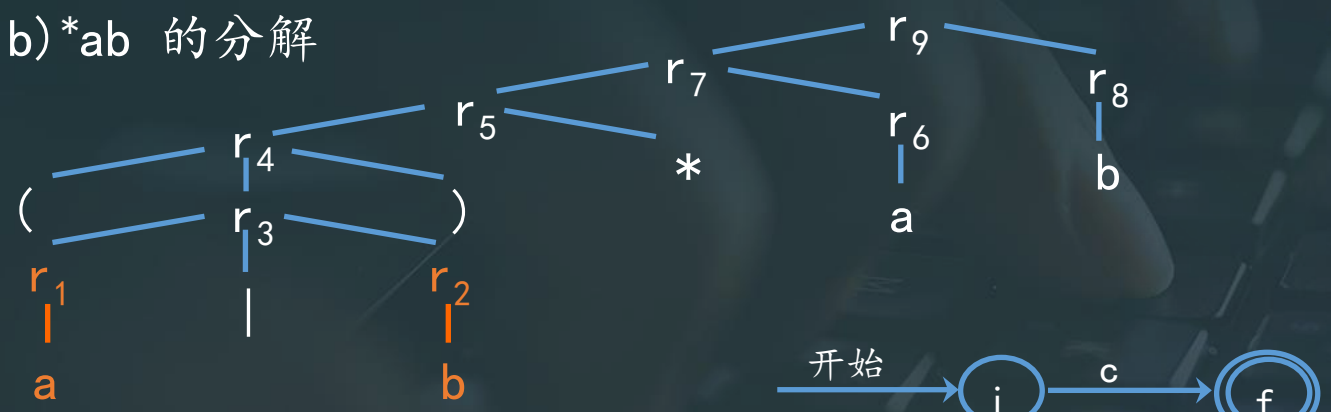
构造 (a|b)*ab 的NFA

(a|b)*ab 的分解

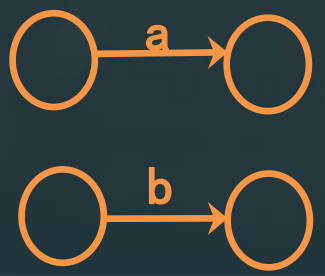


构造 (a|b)*ab 的NFA

(a|b)*ab 的分解

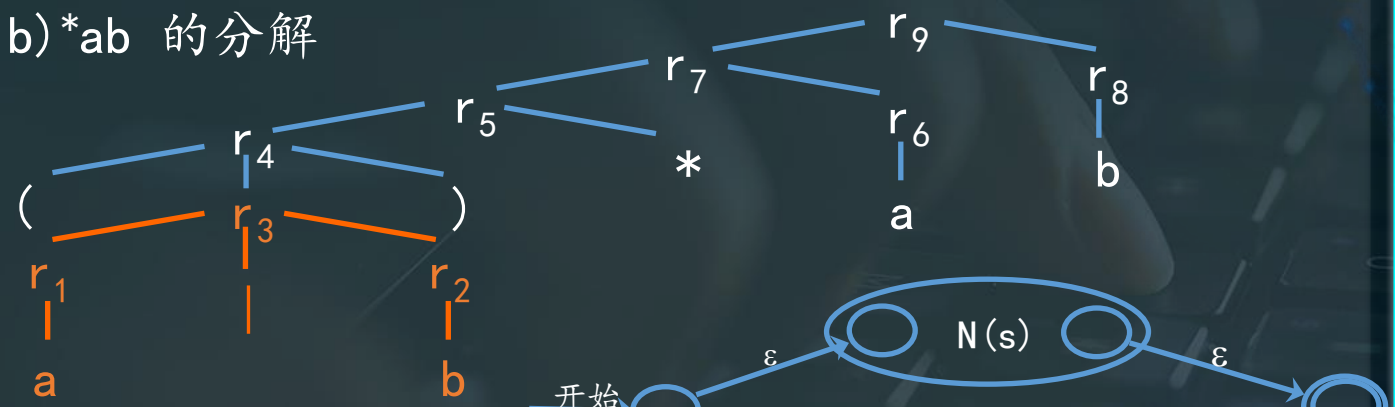


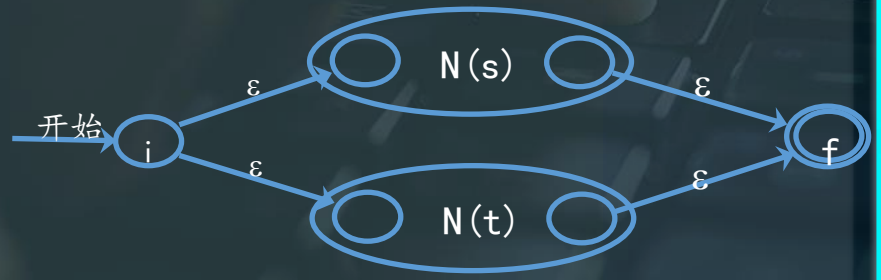
识别正则式 c 的NFA



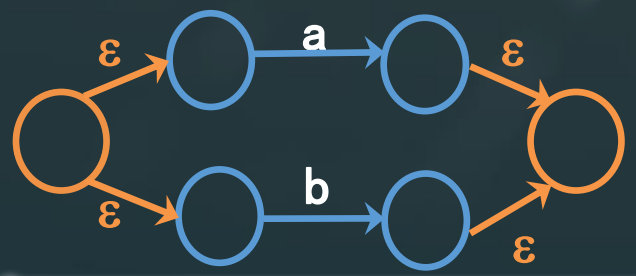
构造 (a|b)*ab 的NFA

(a|b)*ab 的分解



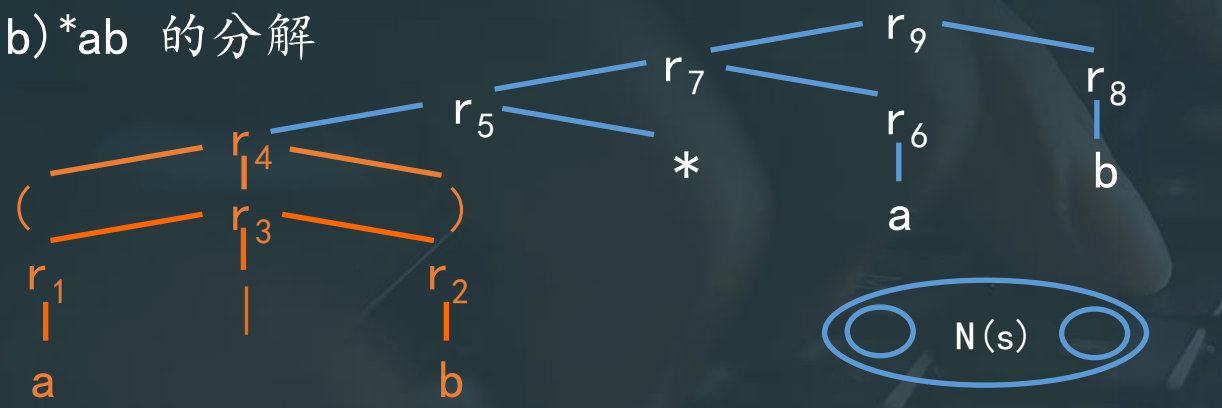


识别正则式 s|t 的NFA

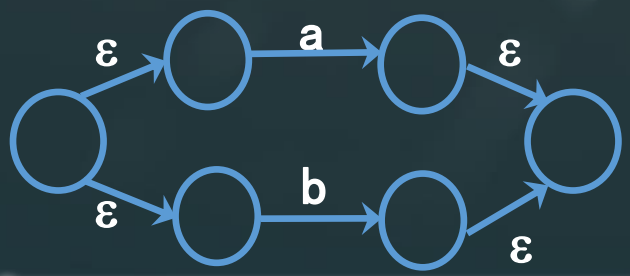


构造 (a|b)*ab 的NFA

(a|b)*ab 的分解

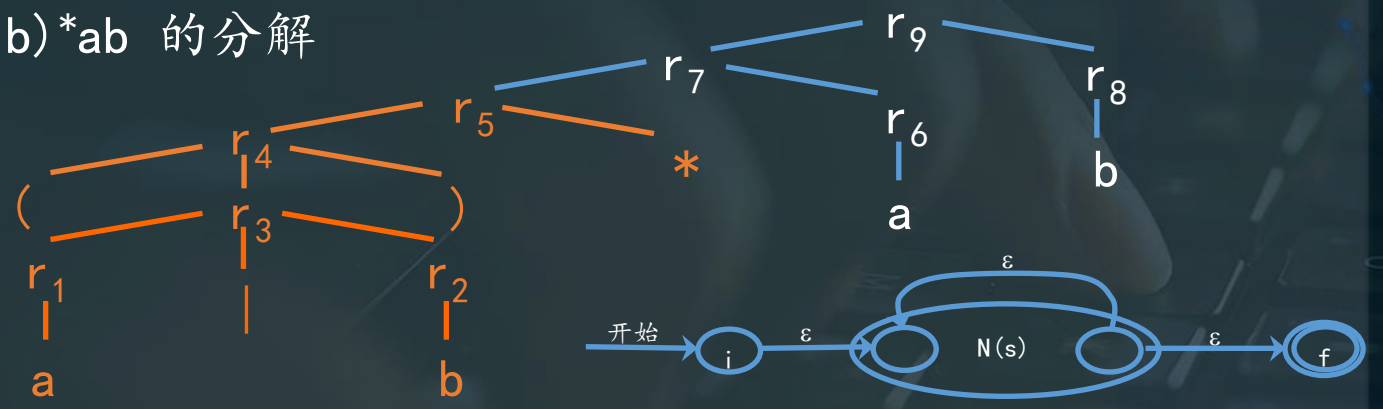


识别正则式（s）的NFA

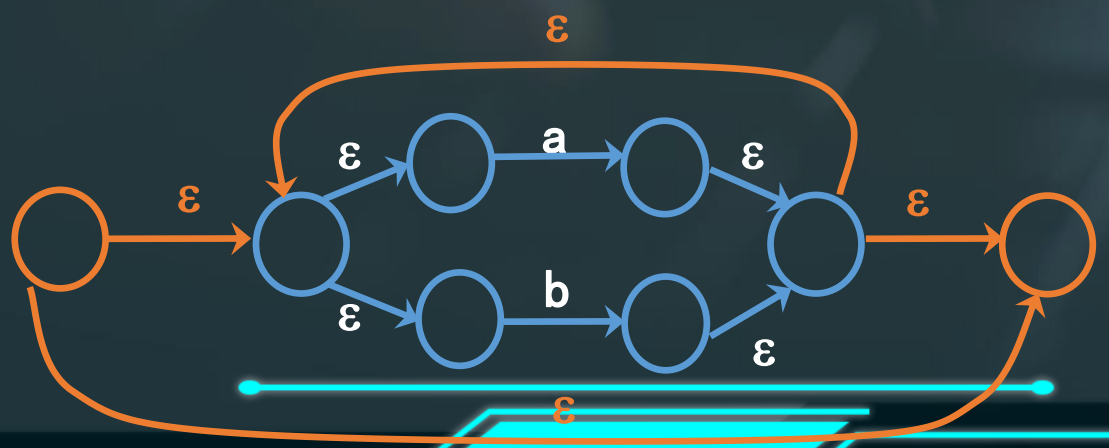


构造 (a|b)*ab 的NFA

(a|b)*ab 的分解

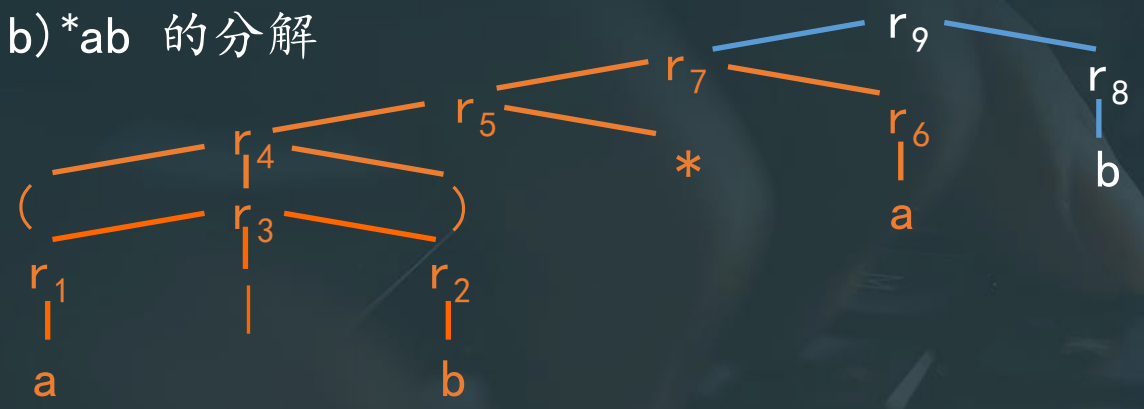


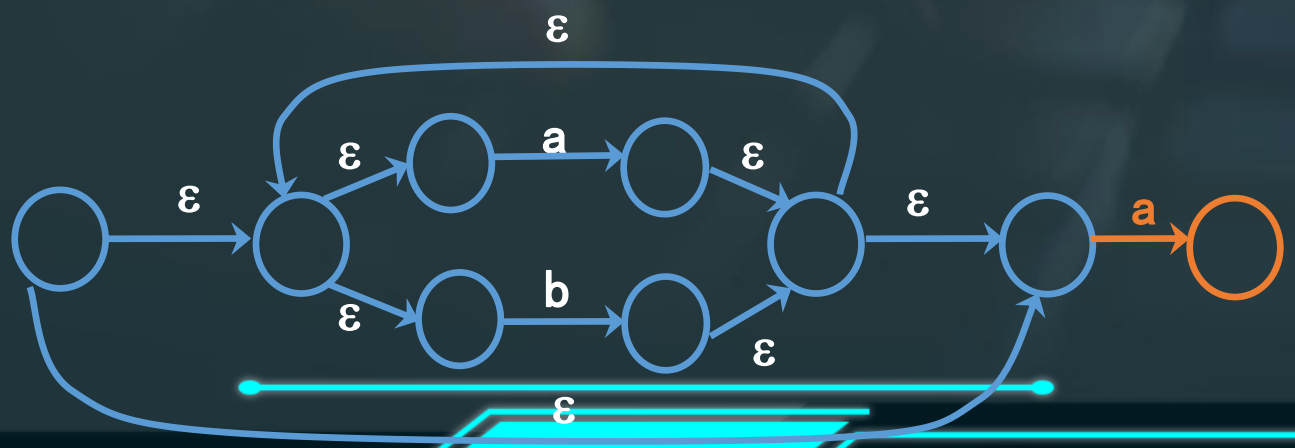
识别正则式 s* 的NFA



构造 (a|b)*ab 的NFA

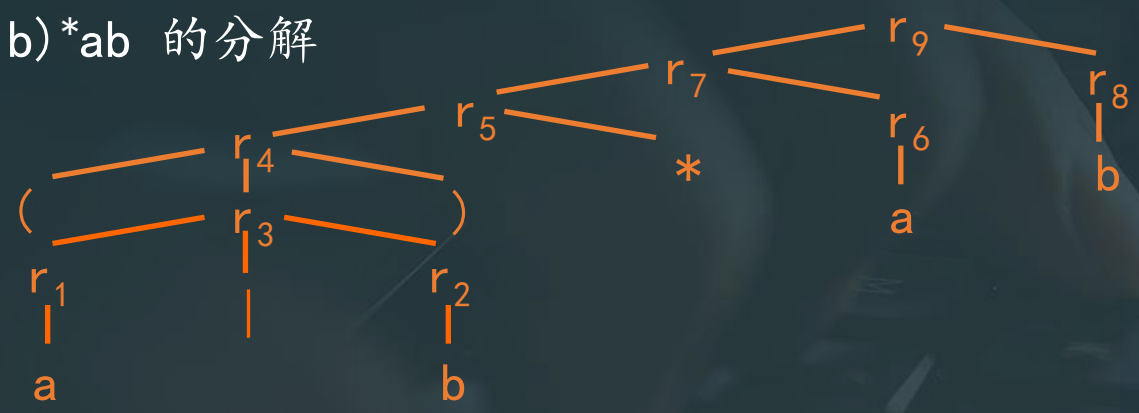
(a|b)*ab 的分解

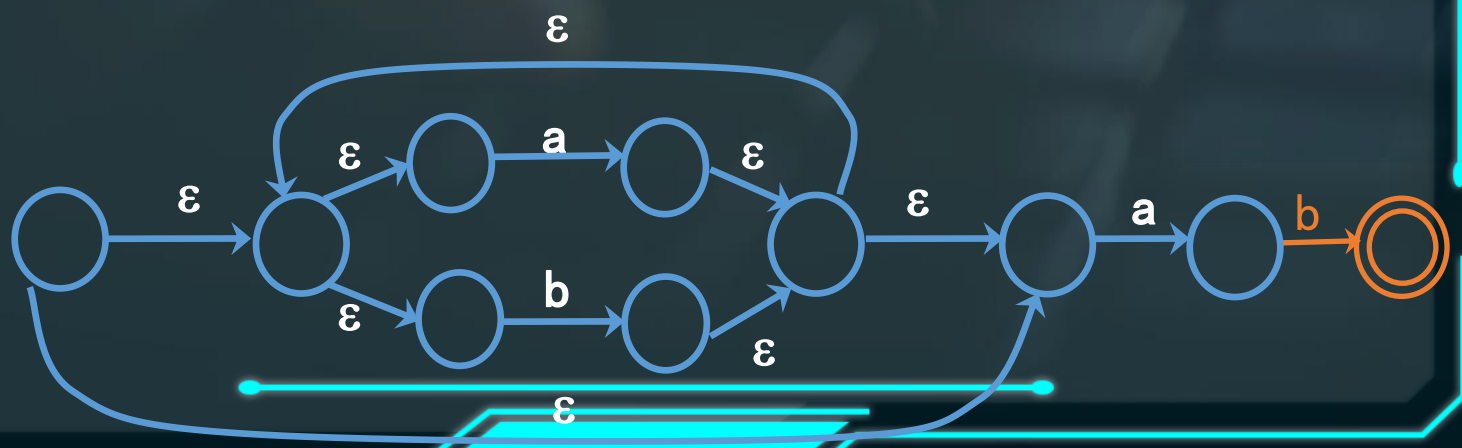




构造 (a|b)*ab 的NFA

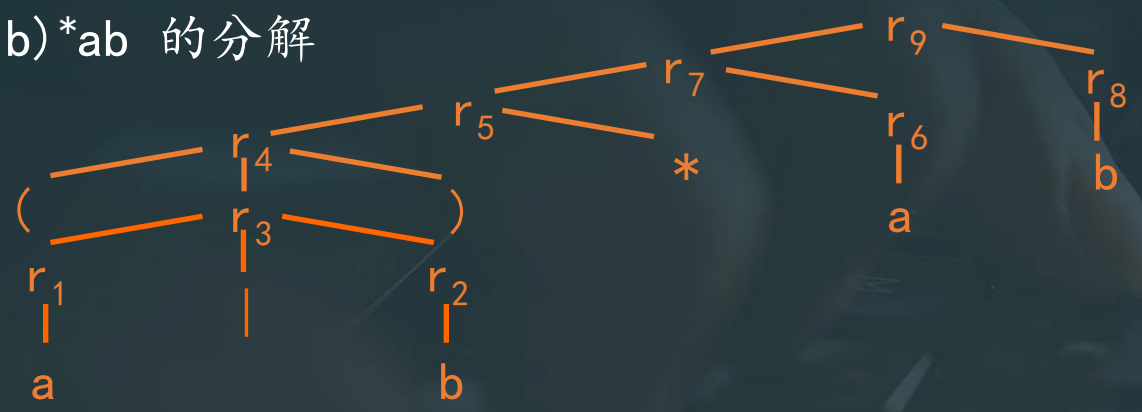
(a|b)*ab 的分解

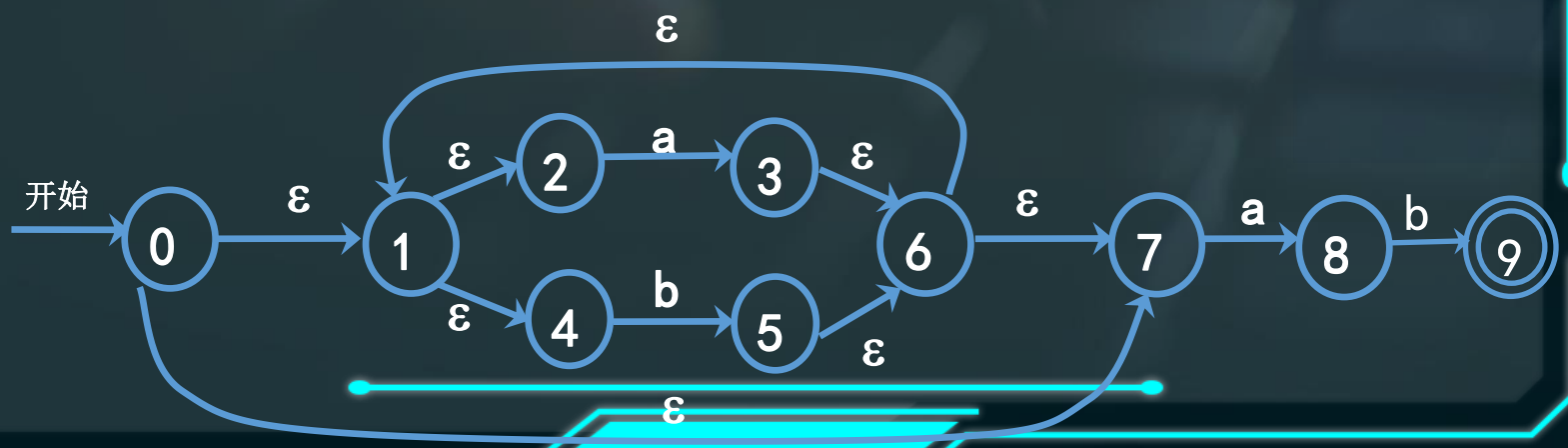




构造 (a|b)*ab 的NFA

(a|b)*ab 的分解





总结

- MYH算法可以将任何正则表达式转变为接受相同语言的NFA。
- 该算法是语法制导的，即它沿着正则表达式的语法分析树自底向上递归的进行处理。
- 对于每个子表达式，该算法构造一个只有一个接受状态的NFA。

编译原理

苏州大学　李军辉