02

OPEN ORIENTED

凹 凸 实 验 室

正则表达式 - 回溯

youshan

问题由来

我们想要从下面的字符串中匹配出所有的“ xxx ”

“ test1 none test2 ”

我们可以通过正则表达式进行匹配，那么简单切入，我们可以写出这样的正则

```
/<div>.+<\/div>/g
```

问题由来

那么通过上述正则，我们能匹配出我们想要的结果吗?

```
var str = ‘<div>test1</div>none<div>test2</div>’

str.match(/<div>.+<\/div>/g)
```

得到的结果只会是:

```
[‘<div>test1</div>none<div>test2</div>’]
```

并非我们期望的:

```
[‘<div>test1</div>’, ‘<div>test2</div>’]
```

量词贪婪模式

这里就涉及到正则表达式中量词的贪婪模式

```
/<div>.+<\/div>/g
```

这里使用了 + 这个量词，表示的含义是 {1, n}，一个或n个。贪婪模式简单理解就是，能匹配我就尽量匹配。那么这个匹配过程又是怎么样的呢？如果在尽量匹配的过程中，遇到了无法匹配的情况（假如上述字符串最后并不是 结尾的），又是怎么处理的呢？

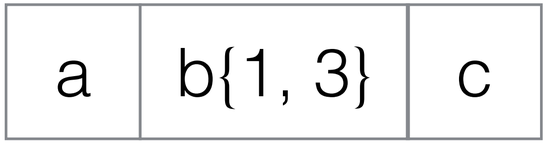
回溯

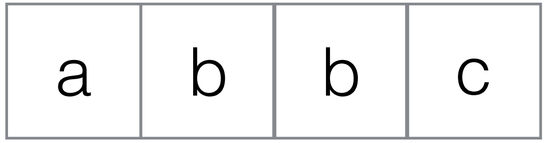
这里涉及到了一个听起来高大上的概念：回溯。

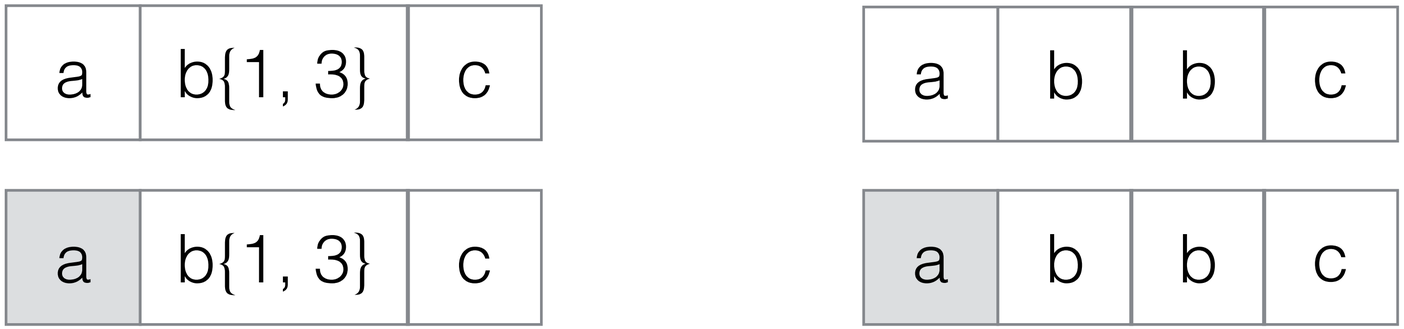
可以理解为在走迷宫的时候，遇到死胡同就掉头往回走一个路口，去另一个方向试试。

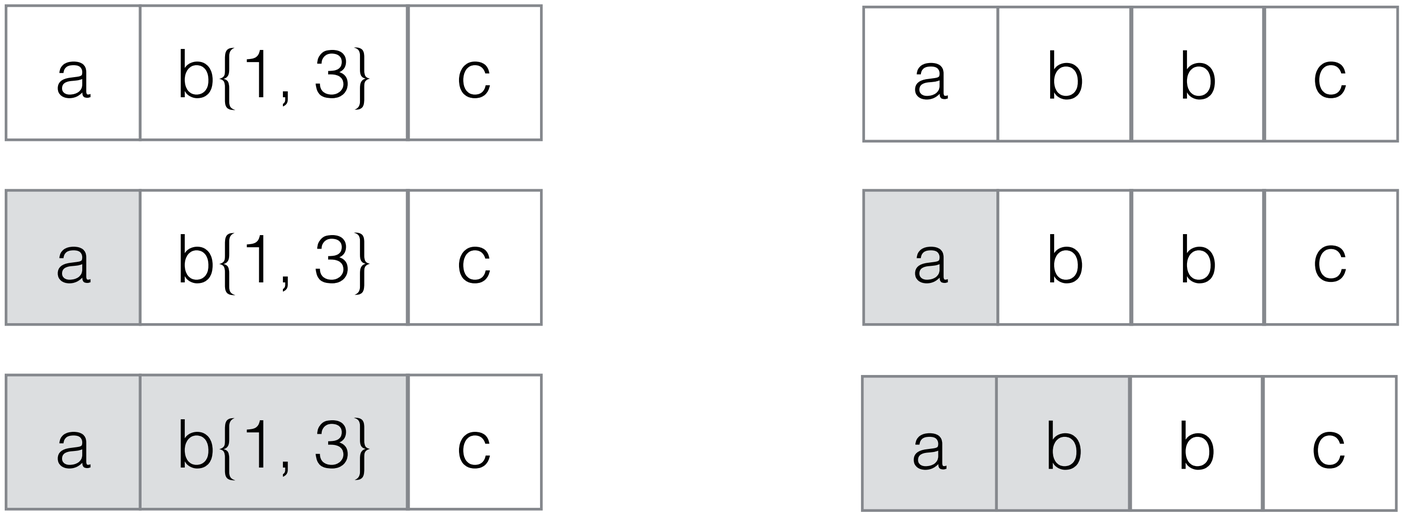
正则表达式：/ab{1, 3}c/

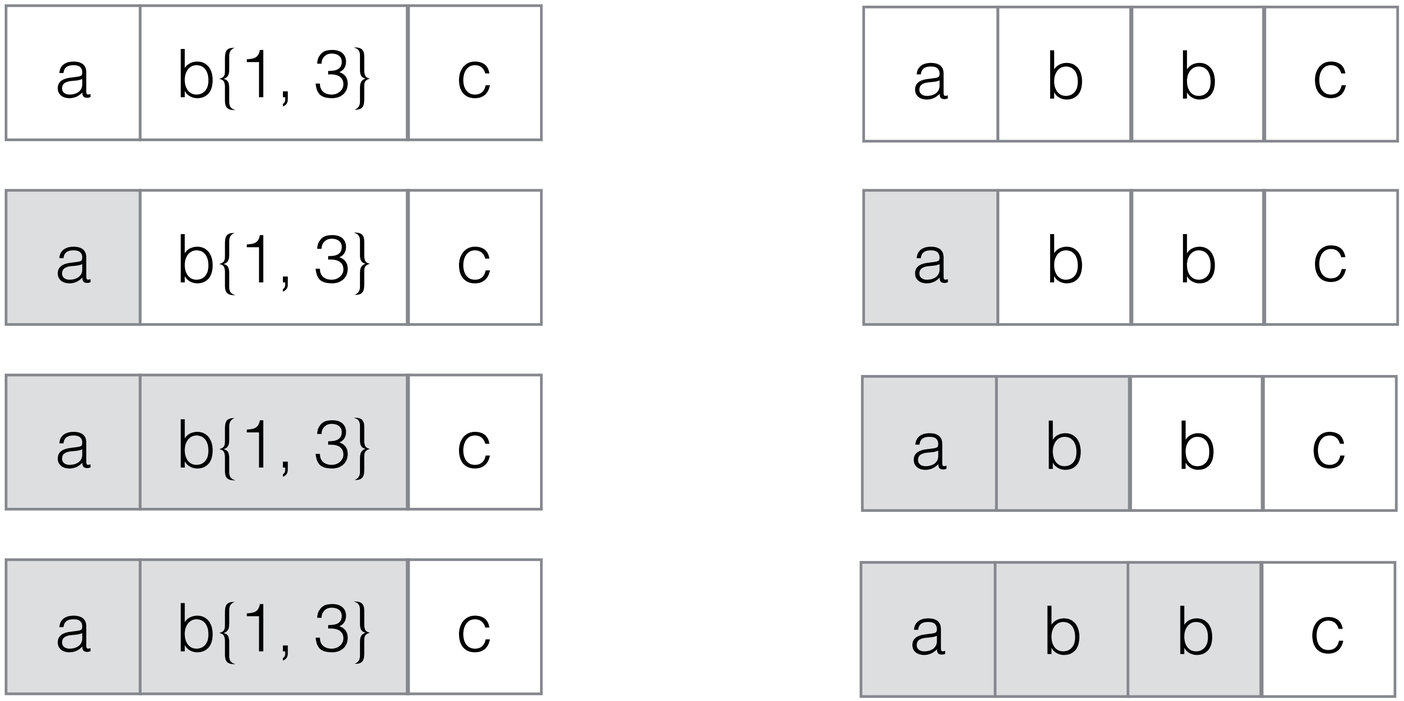
目标字符串：abbc

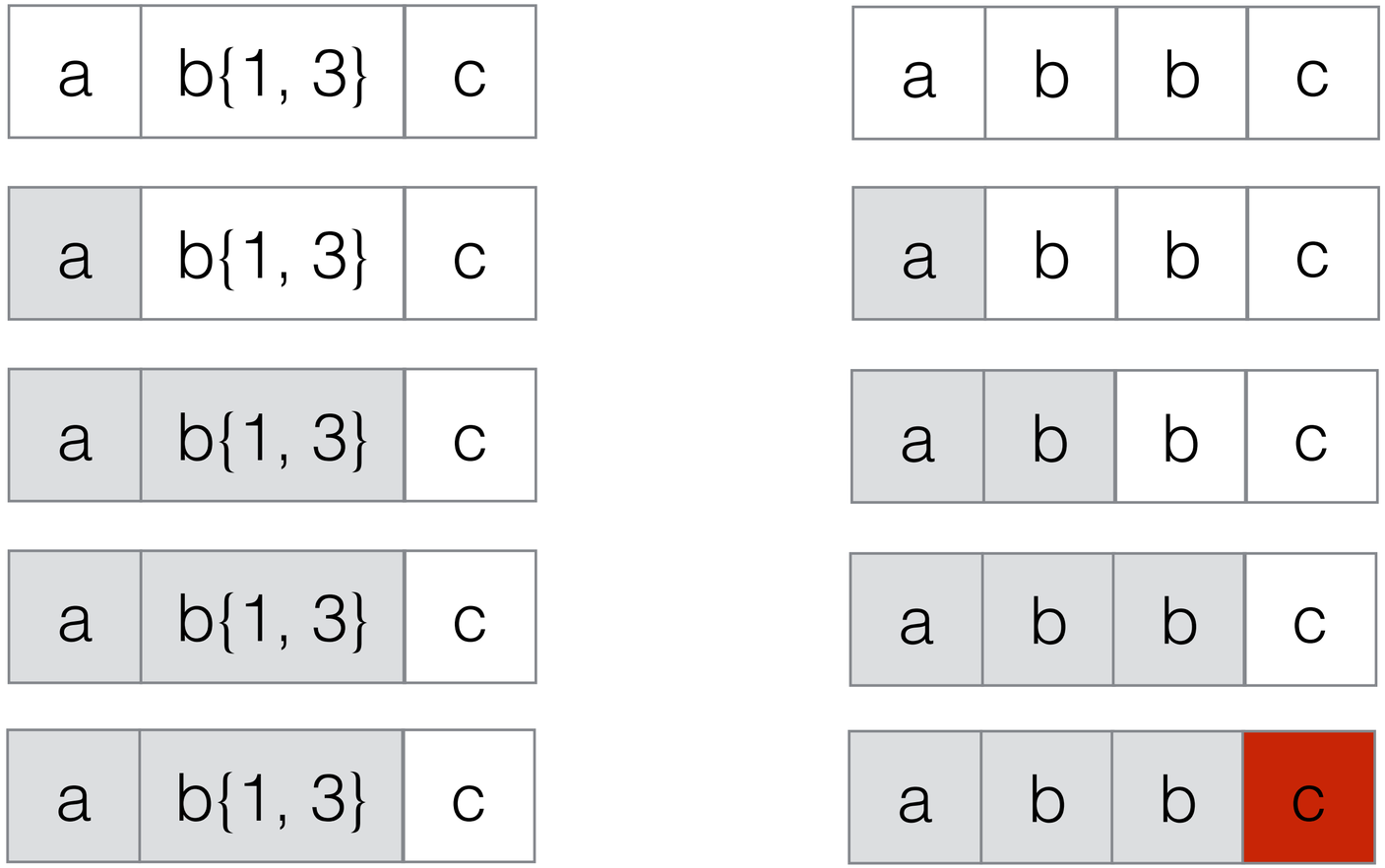

a
b{1, 3}
c

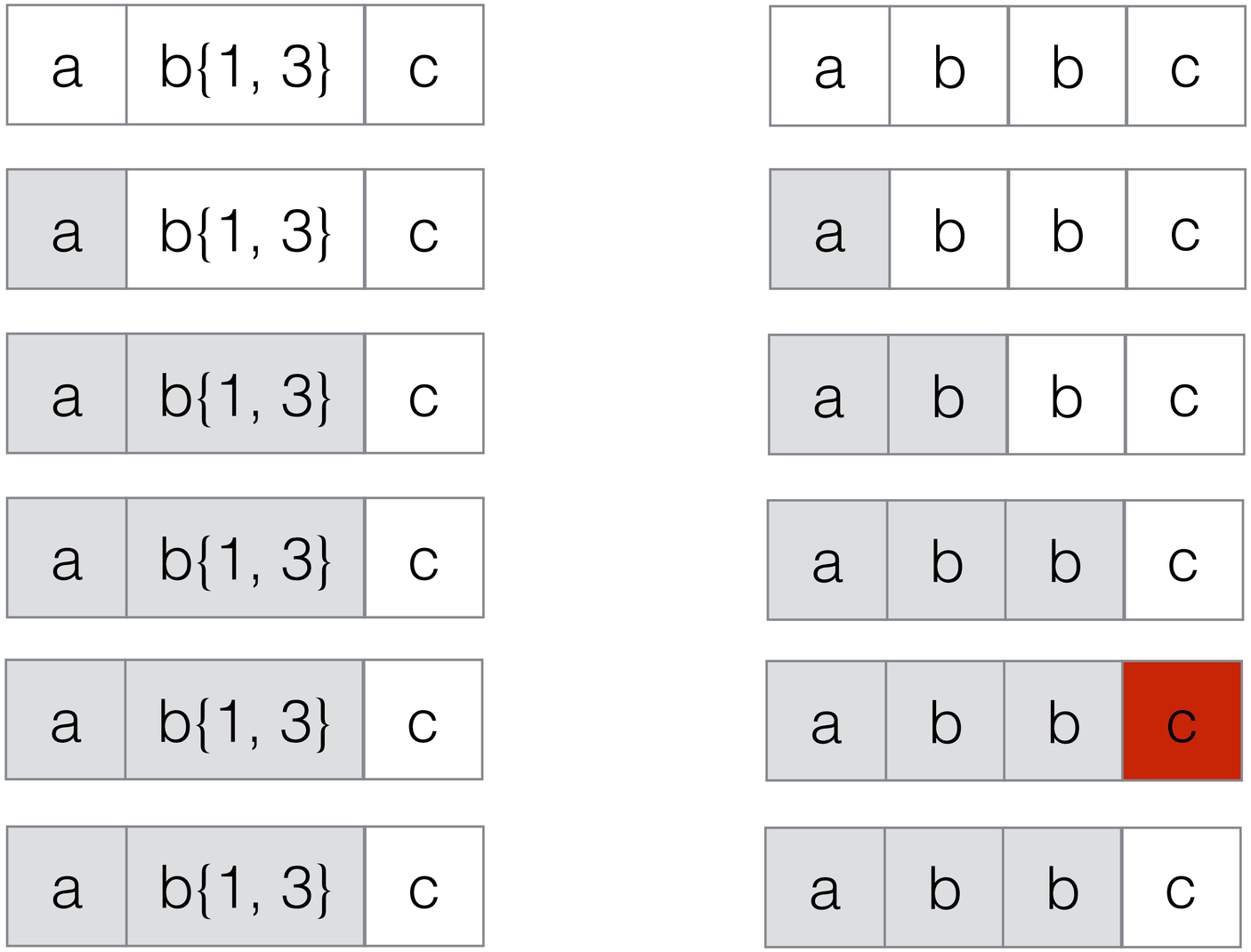

a
b
b
c

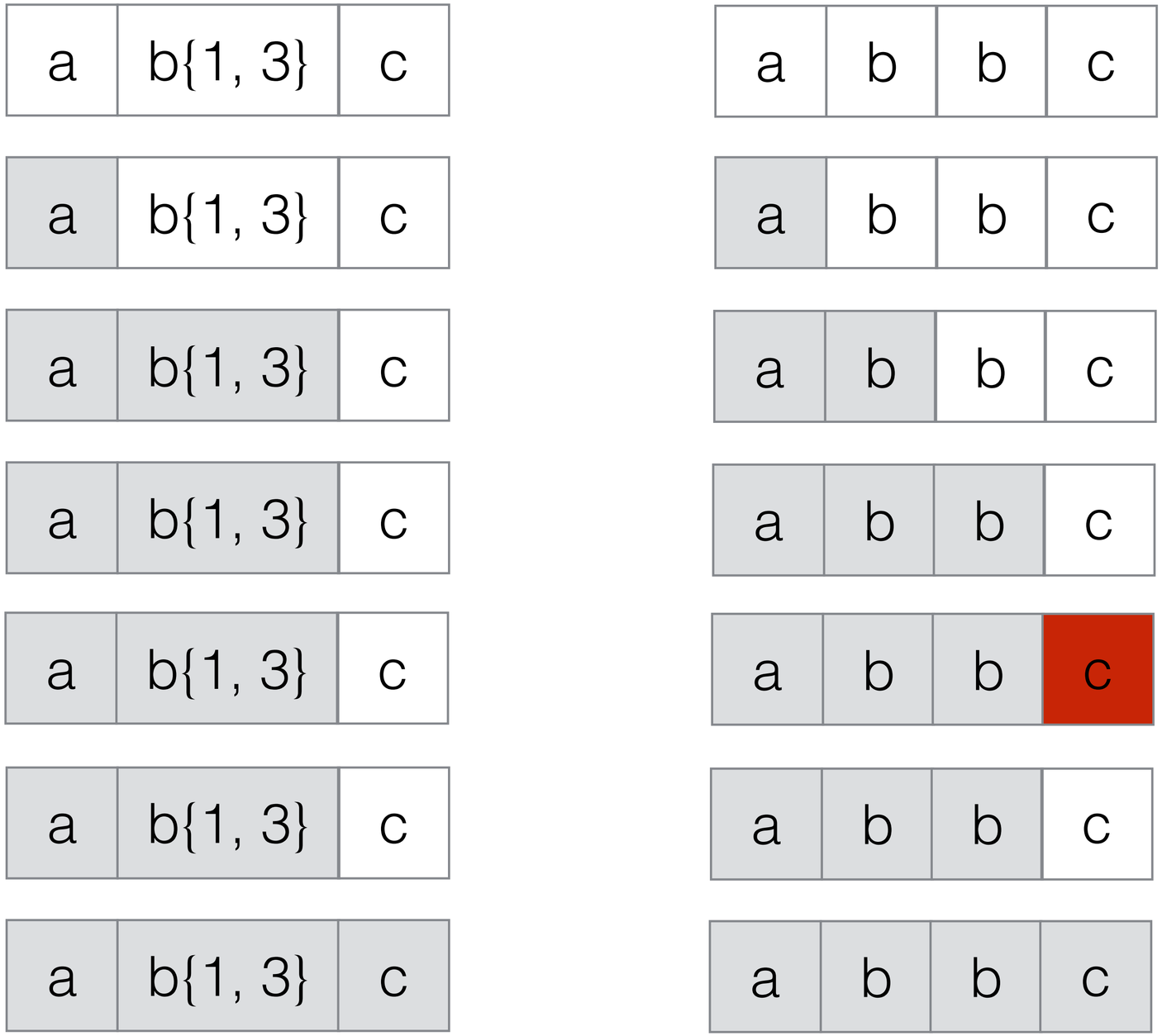

a
b{1, 3}
c
a
b{1, 3}
c
a
b
b
c
a
b
b
c


a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b b c
a b b c
a b b c

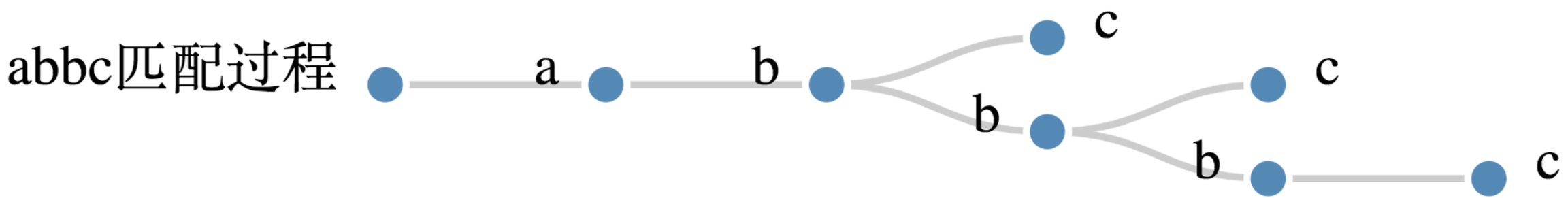

a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b b c
a b b c
a b b c
a b b c


a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b b c
a b b c
a b b c
a b b c
a b b c

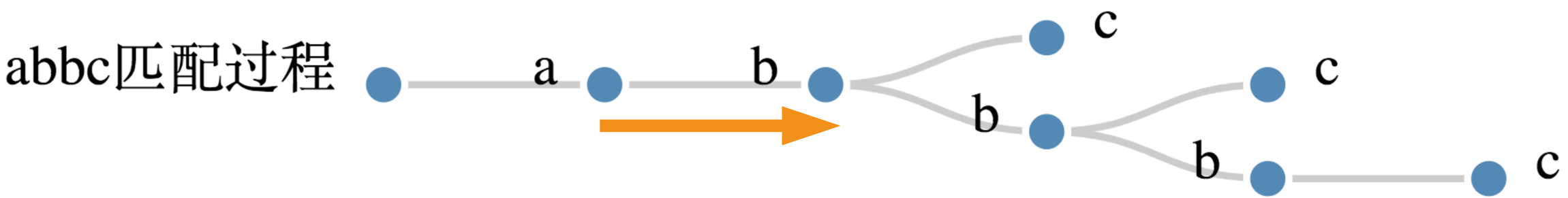

a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b b c
a b b c
a b b c
a b b c
a b b c
a b b c


a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b{1, 3} c
a b b c
a b b c
a b b c
a b b c
a b b c
a b b c
a b b c

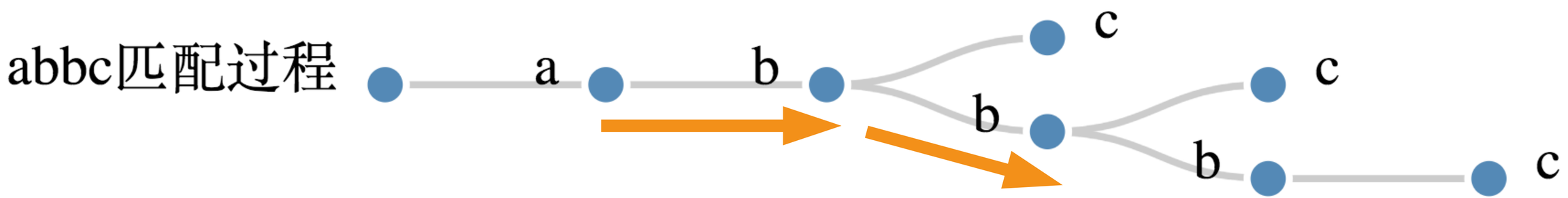
贪婪模式-深度优先的匹配过程



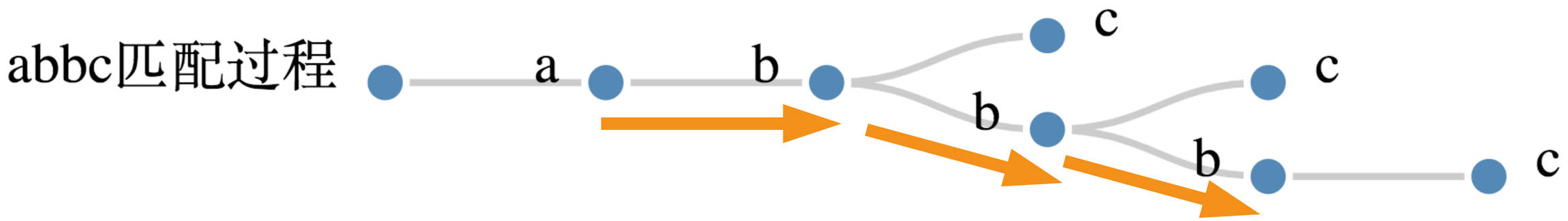
贪婪模式-深度优先的匹配过程



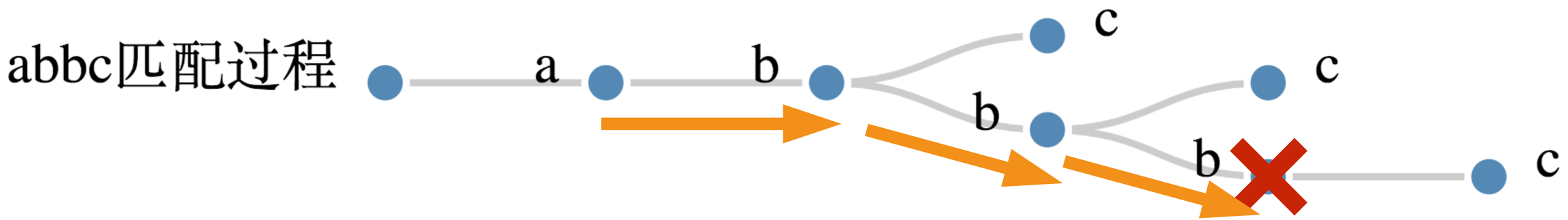
贪婪模式-深度优先的匹配过程



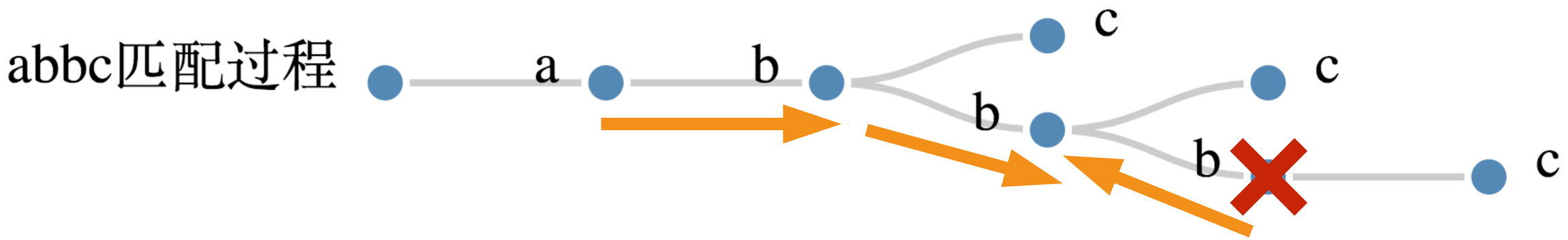
贪婪模式-深度优先的匹配过程



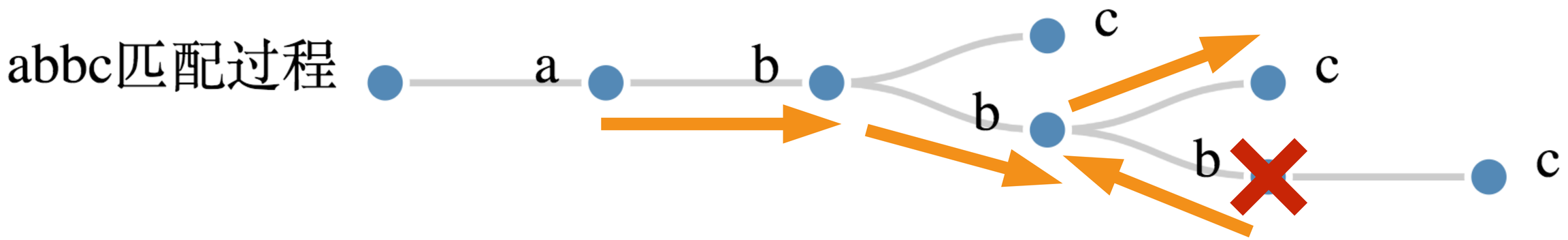
贪婪模式-深度优先的匹配过程



贪婪模式-深度优先的匹配过程



贪婪模式-深度优先的匹配过程



回溯在正则中的应用

1、贪婪模式：尽可能多匹配，多要

2、惰性模式：尽可能少匹配，少要

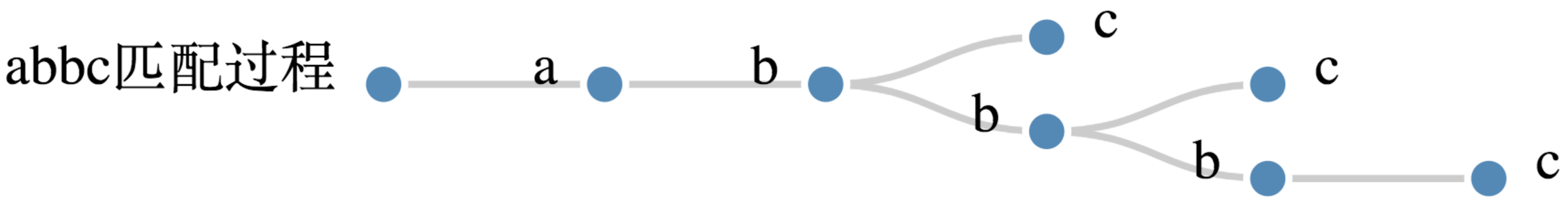
3、分支结构：从左往右依次看，先匹配上哪个就哪个

惰性模式

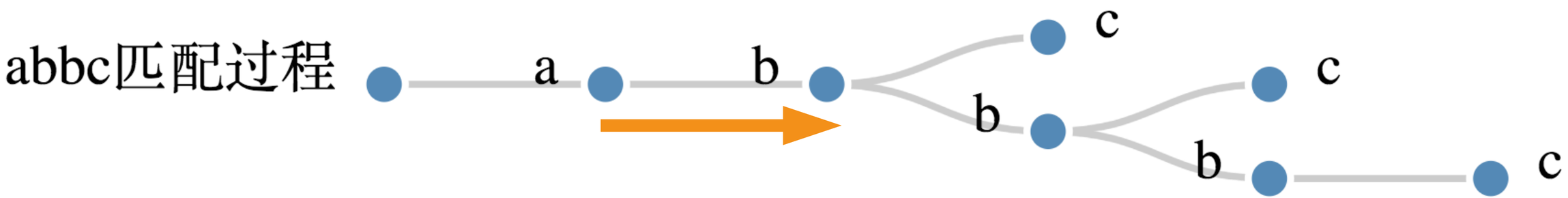
只要在正则表达式的量词后面加上？即可开启惰性匹配模式

```
/<div>.+?<\/div>/g
```

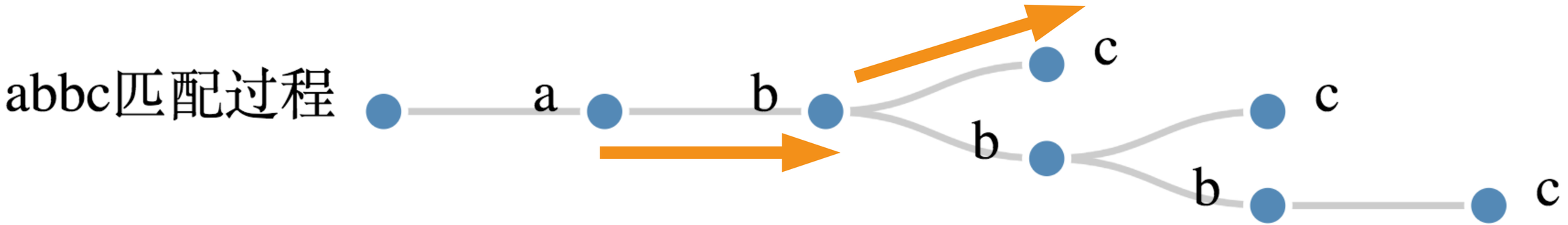
惰性模式-深度优先的匹配过程



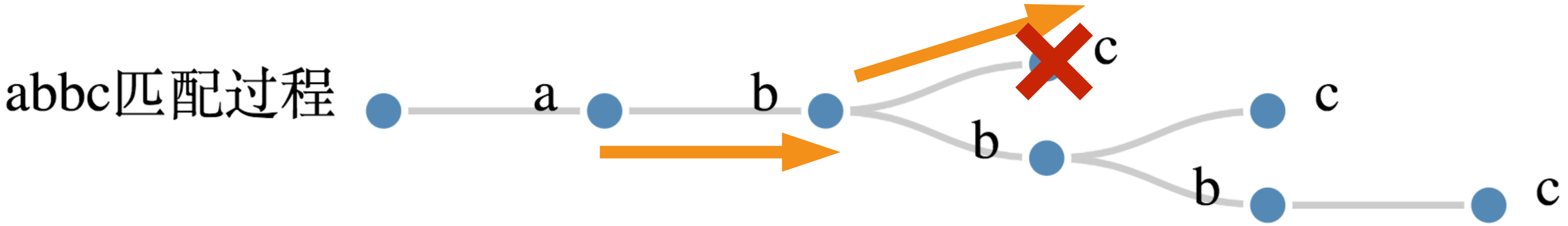
惰性模式-深度优先的匹配过程



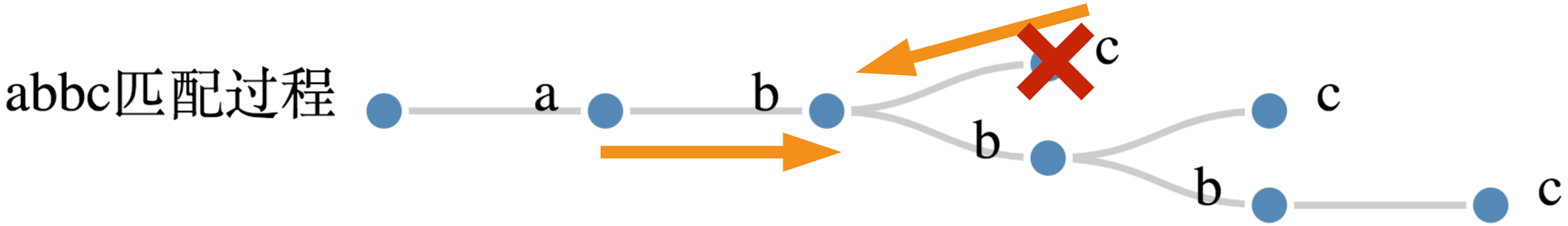
惰性模式-深度优先的匹配过程



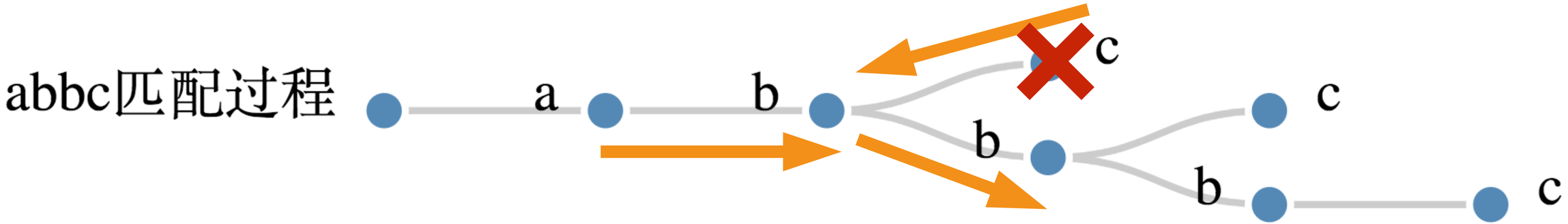
惰性模式-深度优先的匹配过程



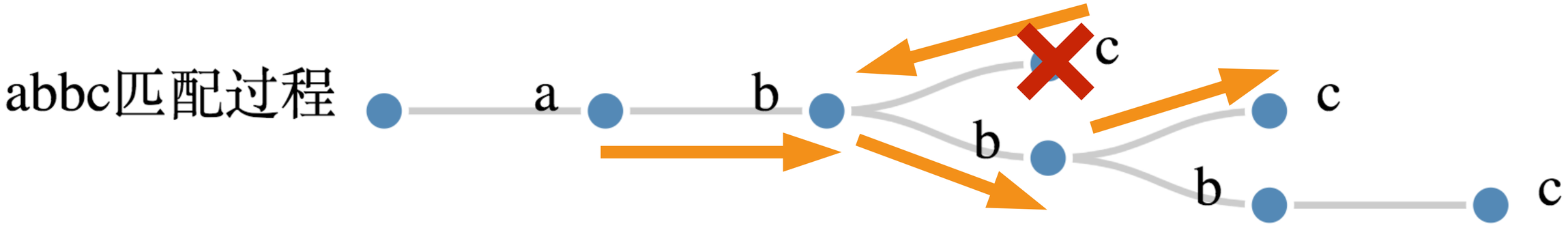
惰性模式-深度优先的匹配过程



惰性模式-深度优先的匹配过程



惰性模式-深度优先的匹配过程



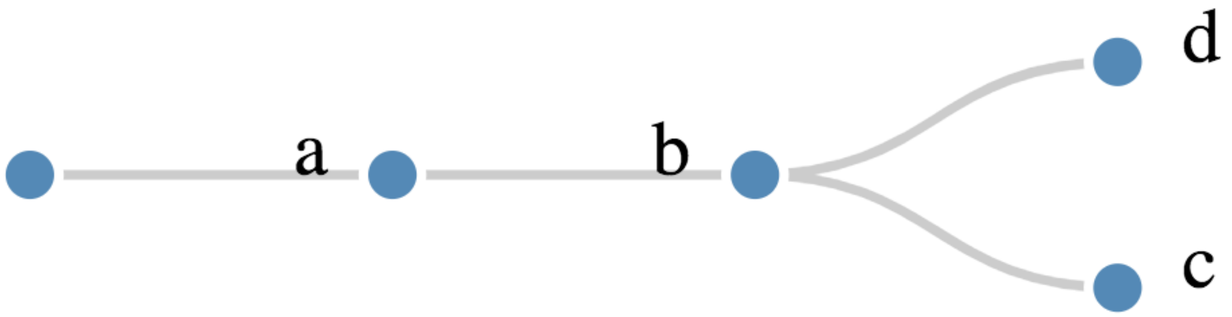
分支模式

分支模式就相对简单了

```
/ab(c|d)/g
```

这里所表达的含义就是要么是 abc 要么是 abd

分支结构匹配



解决最初的问题

问题回顾

那么通过上述正则，我们能匹配出我们想要的结果吗?

```
var str = ‘<div>test1</div>none<div>test2</div>’
```

```
str.match(/<div>.+<\/div>/g)
```

得到的结果只会是:

```
[‘<div>test1</div>none<div>test2</div>’]
```

并非我们期望的:

```
[‘<div>test1</div>’, ‘<div>test2</div>’]
```

只需要开启 惰性模式 即可

```
/<div>.+?<\/div>/g
```

https://regexper.com/

THANKS

FOR YOUR WATCHING

02

OPEN ORIENTED

凹 凸 实 验 室