Regular Expressions / Finite State Automata

UW CSE 490u Quiz Section, Feb 8, 2017

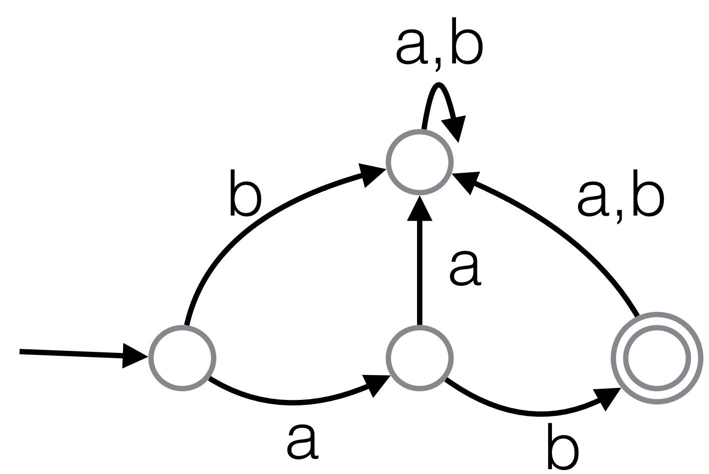
Sam Thomson

Denotations of REs

- $[\![\varnothing]\!] = \varnothing$
- $[\![\varepsilon]\!] = \{\varepsilon\}$
- $[\![\text{ c }]\!] = \{\text{c}\}$ for $\text{c} \in \Sigma$
- $[\![\alpha\,\beta]\!] = \{\, x\,y \mid x \in [\![\alpha]\!],\ y \in [\![\beta]\!] \,\}$
- $[\![\alpha \mid \beta]\!] = [\![\alpha]\!] \cup [\![\beta]\!]$
- $[\![\alpha^*]\!] = \{\, x_0 \ldots x_n \mid x_i \in [\![\alpha]\!],\ n \geq 0 \,\}$

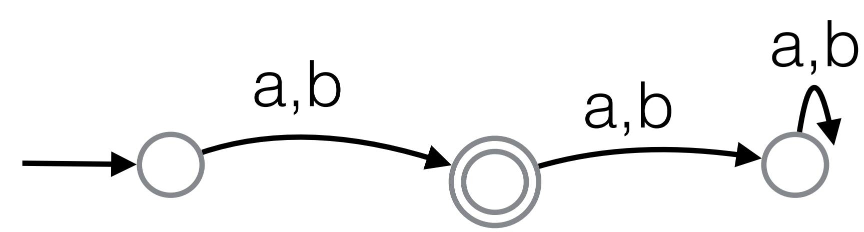
FSA Diagrams

ab



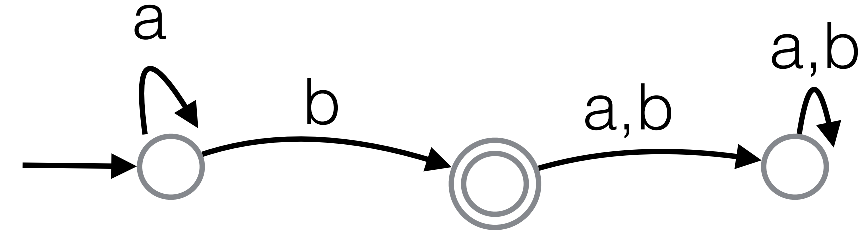
FSA Diagrams

a|b



FSA Diagrams

a*b



Pumping Lemma

Intuition (FSA)

- A regular language L has an FSA F that accepts it.
- F has p states (for some p)
- Any word $w \in L$ with $|w| > p$ will go through the same state twice (pigeonhole), creating a loop
- That loop can be "pumped"

Intuition (RE)

- A regular language L has a RE that accepts it.
- The only way to make an infinite RE is using a "*", e.g. $\alpha\beta^*\gamma|\omega$, with non-empty β
- β can be pumped

Formal Statement

- Let L be a regular language. Then there exists $p \geq 1$ such that every string w in L of length at least p (p is called the "pumping length") can be written as $w = xyz$, satisfying the following conditions:

- $|y| \geq 1$

- $|xy| \leq p$

- for all $i \geq 0$, $xy^iz \in L$

(wikipedia)

Not the only way

- Closure properties
- Myhill–Nerode theorem

The pumping lemma is necessary, not sufficient

- There are non-regular languages that are "pumpable"
- Usually combination of pumping lemma and closure properties is enough

Example

- $L = \{ (ab)^i \mid i \geq 2 \}$
- L is regular
- $L = [\![\text{abab(ab)}^*]\!]$

Example

- $L = \{ a^i b^i \mid i \geq 0 \}$
- L is not regular
- Given p, let $w = a^p b^p$
- Since $|xy| \leq p$, y is only 'a's. Pumping y will lead to more 'a's than 'b's.

Example

- $L = \{ a^{2i} \mid i \geq 0 \}$
- L is regular
- $L = [\![(aa)^*]\!]$

Example

- $L = \{ a^{2^i} \mid i \geq 0 \}$
- L is not regular
- Pumping any $y = a^j$ will work at most once, because the gaps between 2^i double each time.

Example

- $L = \{ ww \mid w \in \{a,b\}^* \}$
- L is not regular
- Given p, let $w = a^p b a^p b$
- Since $|xy| \leq p$, y is only 'a's. Pumping y will lead to more 'a's in the first half than the second half.

Example

- L = { matching "parentheses" } $\subseteq$ {a,b}*
- L is not regular
- Intersect L with $[\![a^*b^*]\!]$ and you get $\{ a^i b^i \mid i \geq 0 \}$.
- We already showed that's not regular

Exercise

- Write CFGs for the previous examples
- For example:
  - $L = \{ a^{2i} \mid i \geq 0 \}$
- Rules:
  - $S \rightarrow aaS$
  - $S \rightarrow \varepsilon$

Write CFGs for:

- $L = \{ a^i b^i \mid i \geq 0 \}$
- $L = \{ (ab)^i \mid i \geq 2 \}$
- $L = \{ a^{2^i} \mid i \geq 0 \}$
- $L = \{ ww \mid w \in \{a,b\}^* \}$
- L = { matching "parentheses" } $\subseteq \{a,b\}^*$

Converting FSAs to REs

(bonus, not on final)

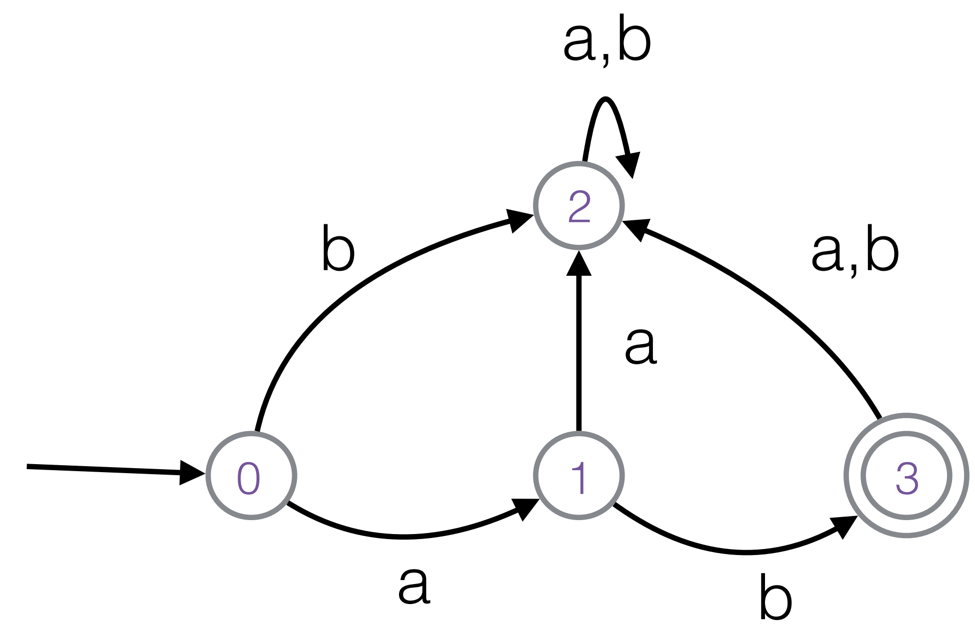
Converting FSAs to REs

- The Floyd-Warshall/Roy/Kleene/Gauss-Jordan/McNaughton/Yamada dynamic programming algorithm computes:
  - transitive closures
  - shortest paths
  - highest probability paths
  - total probabilities of all paths
  - the regular expression for a finite automaton
  - solution to linear equations

Kleene's Algorithm

- Recurrence:
  - $C^{(-1)}$ is adjacency matrix
    - (paths from i to j that don't pass through any other states in between)
  - $C^{(k)}_{i,j} = C^{(k-1)}_{i,j} \mid (C^{(k-1)}_{i,k} (C^{(k-1)}_{k,k})^* C^{(k-1)}_{k,j})$
    - (paths from i to j that possibly pass through 0,...,k)
- Answer is $C_{n,start,final}$

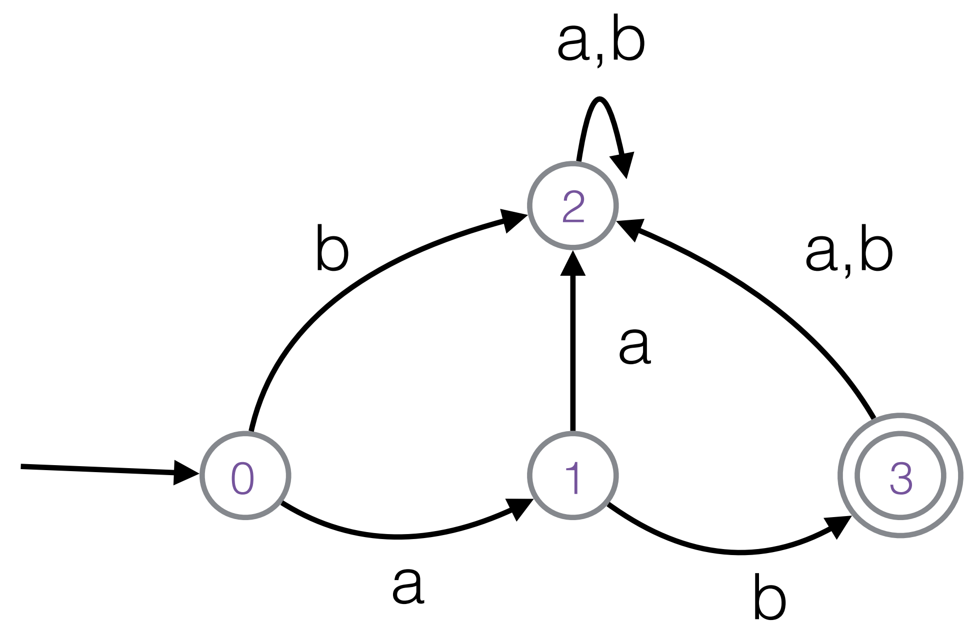
Kleene's Algorithm



Example

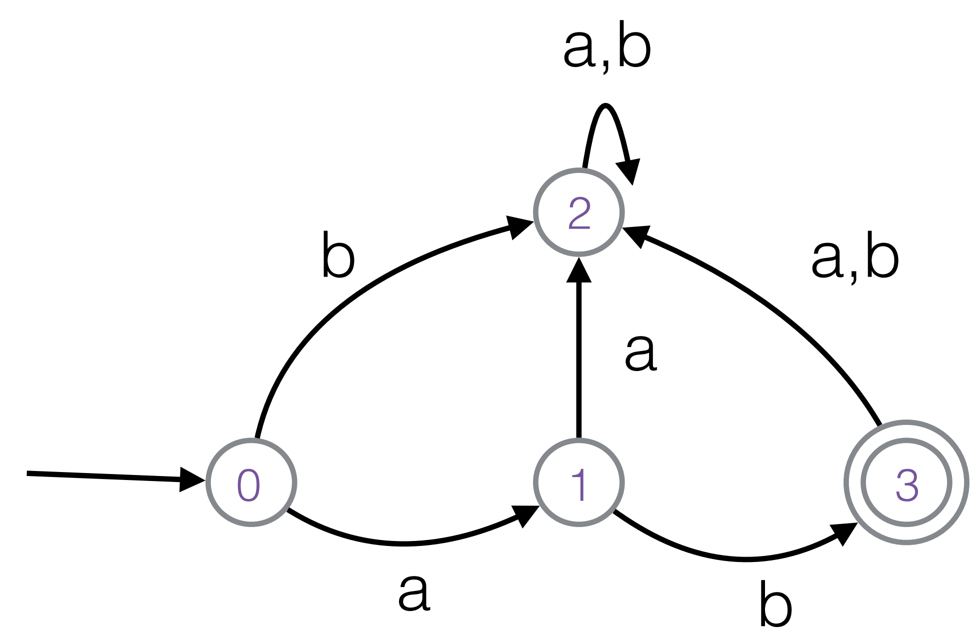
Paths from *i* to *j*
without passing through
any other nodes in between:

$C^{(-1)}$	0	1	2	3
0	∅	a	b	∅
1	∅	∅	a	b
2	∅	∅	a|b	∅
3	∅	∅	a|b	∅



Example

Paths from *i* to *j* that possibly pass through 0

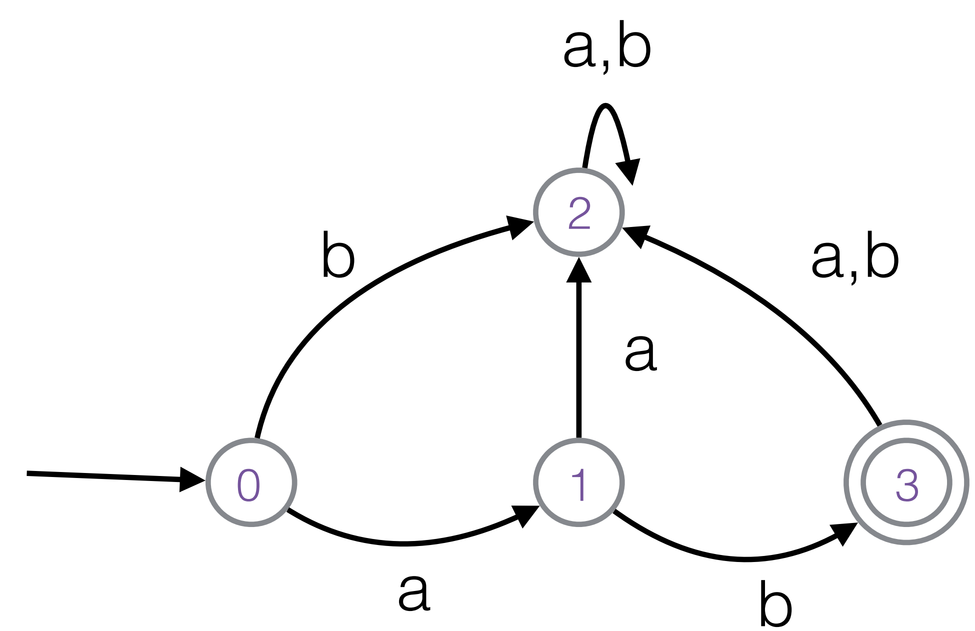


$C^{(-1)}$	0	1	2	3
0	∅	a	b	∅
1	∅	∅	a	b
2	∅	∅	a|b	∅
3	∅	∅	a|b	∅

$C^{(0)}$	0	1	2	3
0	∅	a	b	∅
1	∅	∅	a	b
2	∅	∅	a|b	∅
3	∅	∅	a|b	∅

Example

Paths from *i* to *j* that possibly pass through 0,1

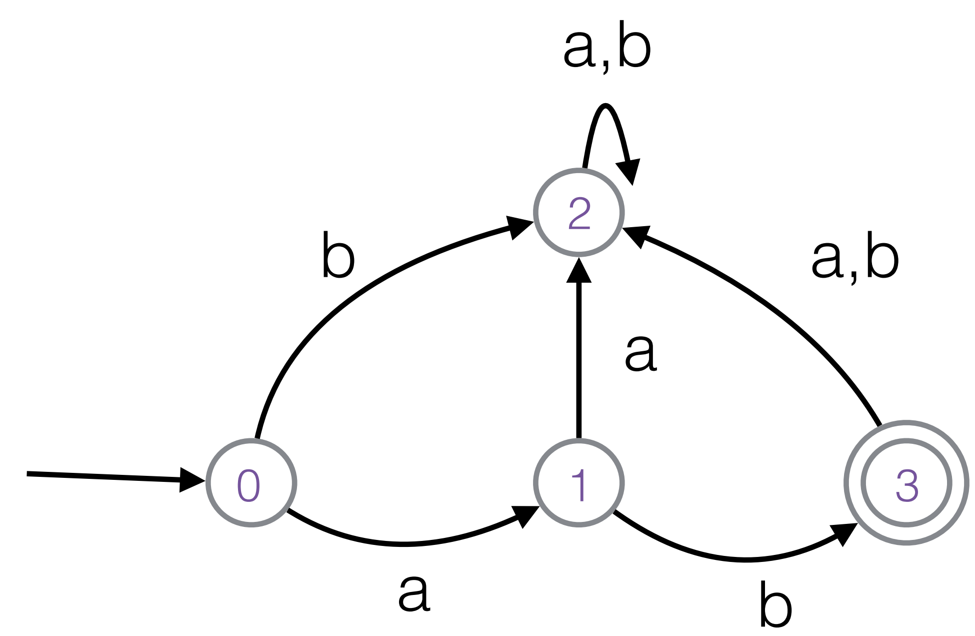


C(0)	**0**	**1**	**2**	**3**
0	∅	a	b	∅
1	∅	∅	a	b
2	∅	∅	a|b	∅
3	∅	∅	a|b	∅

C(1)	**0**	**1**	**2**	**3**
0	∅	a	b|aa	ab
1	∅	∅	a	b
2	∅	∅	a|b	∅
3	∅	∅	a|b	∅

Example

Paths from *i* to *j* that possibly pass through 0,1,2

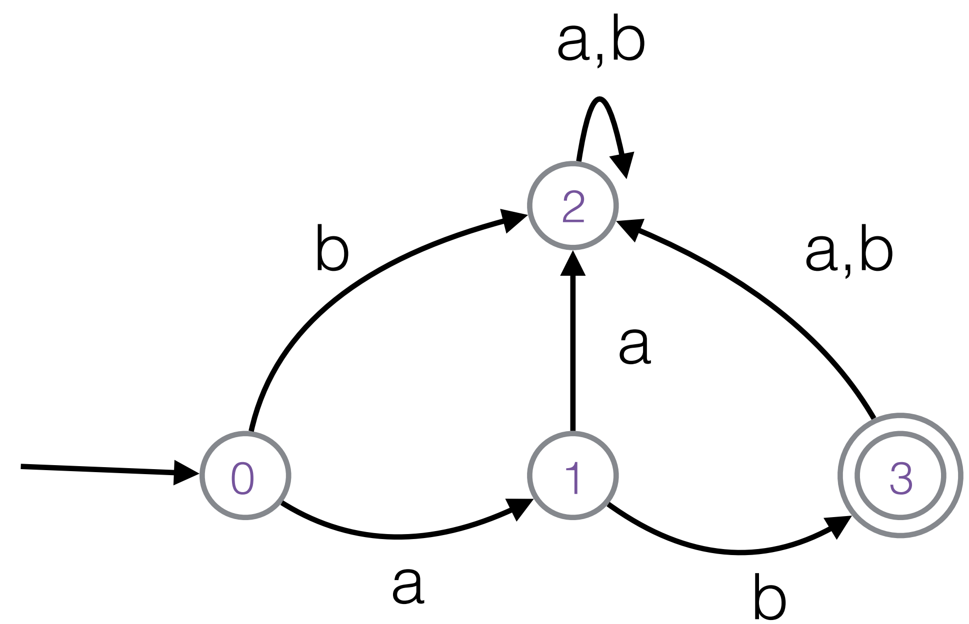


C(1)	0	1	2	3
0	∅	a	b|aa	ab
1	∅	∅	a	b
2	∅	∅	a|b	∅
3	∅	∅	a|b	∅

C(2)	0	1	2	3
0	∅	a	(b|aa)(a|b)*	ab
1	∅	∅	a(a|b)*	b
2	∅	∅	(a|b)*	∅
3	∅	∅	(a|b)(a|b)*	∅

Example

Paths from *i* to *j* that possibly pass through 0,1,2,3

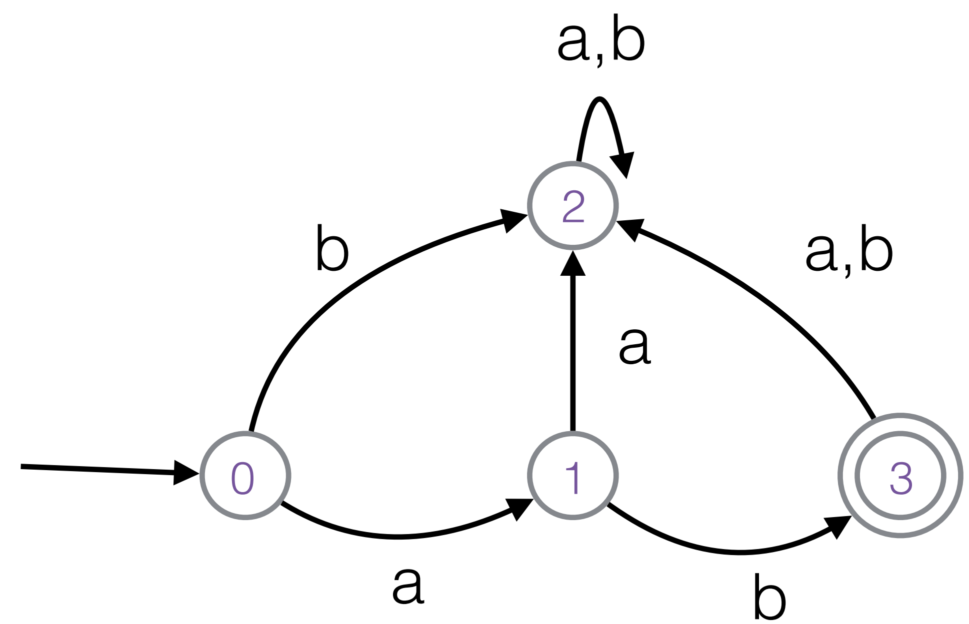


<table>
<tr><th>C(2)</th><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>0</td><td>∅</td><td>a</td><td>(b|aa)(a|b)*</td><td>ab</td></tr>
<tr><td>1</td><td>∅</td><td>∅</td><td>a(a|b)*</td><td>b</td></tr>
<tr><td>2</td><td>∅</td><td>∅</td><td>(a|b)*</td><td>∅</td></tr>
<tr><td>3</td><td>∅</td><td>∅</td><td>(a|b)(a|b)*</td><td>∅</td></tr>
</table>

<table>
<tr><th>C(3)</th><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>0</td><td>∅</td><td>a</td><td>((b|aa)(a|b)*)|(ab (a|b)(a|b)*)</td><td>ab</td></tr>
<tr><td>1</td><td>∅</td><td>∅</td><td>(a(a|b)*)|(b (a|b)(a|b)*)</td><td>b</td></tr>
<tr><td>2</td><td>∅</td><td>∅</td><td>(a|b)*</td><td>∅</td></tr>
<tr><td>3</td><td>∅</td><td>∅</td><td>(a|b)(a|b)*</td><td>∅</td></tr>
</table>

Example

Paths from *i* to *j* that possibly pass through 0,1,2,3



<table>
<tr><th>C(2)</th><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>0</td><td>∅</td><td>a</td><td>(b|aa)(a|b)*</td><td>ab</td></tr>
<tr><td>1</td><td>∅</td><td>∅</td><td>a(a|b)*</td><td>b</td></tr>
<tr><td>2</td><td>∅</td><td>∅</td><td>(a|b)*</td><td>∅</td></tr>
<tr><td>3</td><td>∅</td><td>∅</td><td>(a|b)(a|b)*</td><td>∅</td></tr>
</table>

<table>
<tr><th>C(3)</th><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>0</td><td>∅</td><td>a</td><td>((b|aa)(a|b)*)|(ab (a|b)(a|b)*)</td><td>ab</td></tr>
<tr><td>1</td><td>∅</td><td>∅</td><td>(a(a|b)*)|(b (a|b)(a|b)*)</td><td>b</td></tr>
<tr><td>2</td><td>∅</td><td>∅</td><td>(a|b)*</td><td>∅</td></tr>
<tr><td>3</td><td>∅</td><td>∅</td><td>(a|b)(a|b)*</td><td>∅</td></tr>
</table>