1 Einleitung

Reguläre Ausdrücke

- Was ist das ?
- Tut das weh?
- Kann man etwas dagegen machen

Motivation

Werkzeuge

- find
- grep
- sed
- awk

Standards

- ERE
- BRE
- PCRE

2 Ein wenig Theorie

Worum geht's?

wichtige Begriffe

- Sprache {Aal, Aas, Abakus, …, Zylinder, Zyniker, Zynismus}
- Wort Abakus
- Alphabet Σ = {a, b, …, z, a, ö, ü, ß, A, B, …, Z, Ä, Ö, Ü}

Wortmengen

Beschreibung Σ_t= {a, b, c, d, e, f, g, h, 1, 2, 3, 4, 5, 6, 7, 8}

- extensional {a1, a2, a3, …, b1, b2, …, h7, h8}
- intensional {*xy* | *x* ∈ {a, …, h}, *y* ∈ {1, …, 8}}
- reguläre Ausdrücke [a-h][1-8]

2.1 Syntax

Basisregeln

$$S \rightarrow \emptyset$$
$$S \rightarrow \varepsilon$$

$$S \rightarrow a$$

Rekursive Regeln

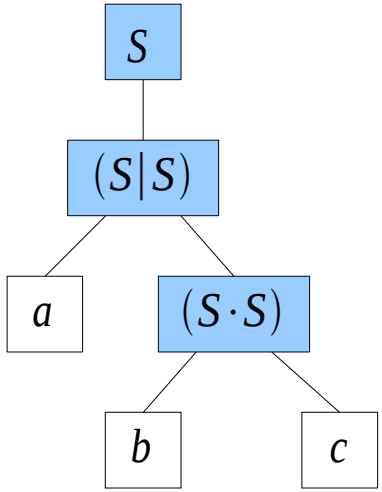
$$S \rightarrow (S \cdot S)$$
$$S \rightarrow (S|S)$$
$$S \rightarrow (S*)$$

$$S \rightarrow (S)$$

Beispiel

$$(a|(b \cdot c))$$

$$S \rightarrow (S|S) \rightarrow (a|S) \rightarrow (a|(S \cdot S))$$
$$\rightarrow (a|(b \cdot S)) \rightarrow (a|(b \cdot c))$$



… nur was hat das eigentlich alles zu bedeuten?

2.2 Semantik

Basisregeln

$$L(\emptyset) := \emptyset$$
$$L(\varepsilon) := \{\varepsilon\}$$

$$L(a) := \{a\}$$

Rekursive Regeln

$$L(R \cdot S) := \{rs \mid r \in L(R),\ s \in L(S)\}$$
$$L(R|S) := L(R) \cup L(S)$$
$$L(S*) := \bigcup_{i=0}^{\infty} L(R^i)$$

$$L((S)) := L(S)$$

Beispiel

$$L(a|(b \cdot c)) =$$
$$= L(a) \cup L((b \cdot c))$$
$$= \{a\} \cup \{rs \mid r \in L(b),\ s \in L(c)\}$$
$$= \{a\} \cup \{rs \mid r \in \{b\},\ s \in \{c\}\}$$
$$= \{a\} \cup \{bc\}$$
$$= \{a, bc\}$$

2.3 Klammern „sparen“

Vorrangregeln

(von stark nach schwach)

- `*` Kleen'sche Hülle
- `•` Verkettung
- `|` Alternative

Beispiele

$$(a|(b\cdot c)) = a|bc$$

$$((a|b)\cdot c) = (a|b)c$$

$$(a|(b\cdot(c*))) = a|bc*$$

$$(a|((b\cdot c)*)) = a|(bc)*$$

Zeit für ein erstes Experiment!

Was trifft ...

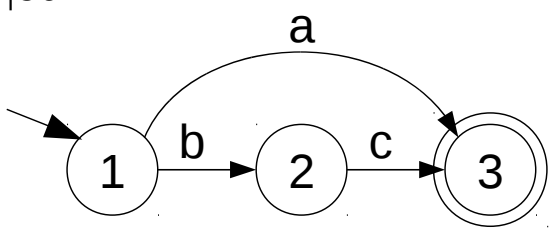
- grep -Ew "a|bc"
- grep -Ew "a|bc*"
- grep -Ew "a|bc*"
- grep -Ew "a|(bc)*"

… und wie beschreibt man $\{a^n b a^n | n \in \mathbb{N}\}$?

2.4 Implementierung

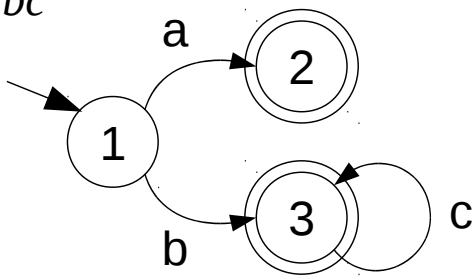
Endliche Automaten

$a|bc$



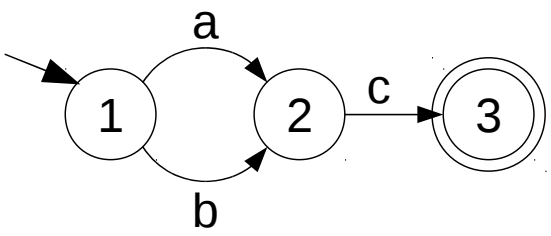
	a	b	c
1	3	2	-
2	-	-	3
3	-	-	-

$a|bc*$



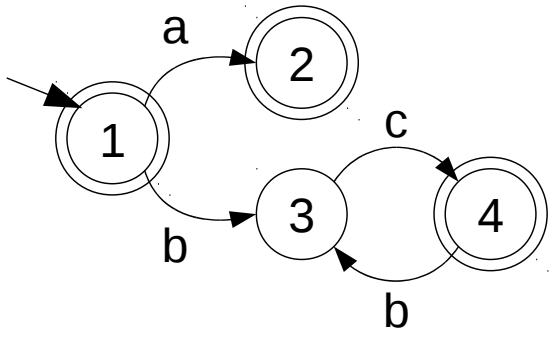
	a	b	c
1	2	3	-
2	-	-	-
3	-	-	3

$(a|b)c$



	a	b	c
1	2	2	-
2	-	-	3
3	-	-	-

$a|(bc)*$



	a	b	c
1	2	3	-
2	-	-	-
3	-	-	4
4	-	-	3

2.5 Abkürzungen

Zeichenklassen

- […] eines der Zeichen …
- [^ …] keines der Zeichen …
- [a - z] ein Zeichen im „Bereich“ a – z

Wiederholungen

- * beliebig oft
- + mindestens ein Mal
- ? höchstens ein Mal
- {m} genau m-Mal
- {m,} mindestens m-Mal
- {m,n} mindestens m, höchstens n-Mal

2.6 Erweiterungen

Anker

gegen implizites .* … .*

- ^ Zeilenanfang
- $ Zeilenende

Rückwärts - Referenz

- \n Treffer des n-ten Teilausdrucks

3 Standards

- POSIX – BRE Basic Regular Expressions
- POSIX – ERE Extended Regular Expressions
- PCRE Perl Compatible Regular Expressions

Beim Lesen unbedingt beachten!

Interpretation eines Zeichens hängt ab von

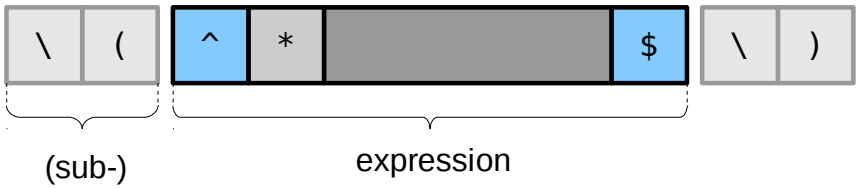
- Kontext (bracket expression […])
- Position

verschiedene Arten vonZeichen

- gewöhnliche Zeichen
- spezielle Zeichen
- „spezielle“ gewöhnliche Zeichen

3.1 POSIX – Basic Regular Expressions (BRE)

Regular Expression



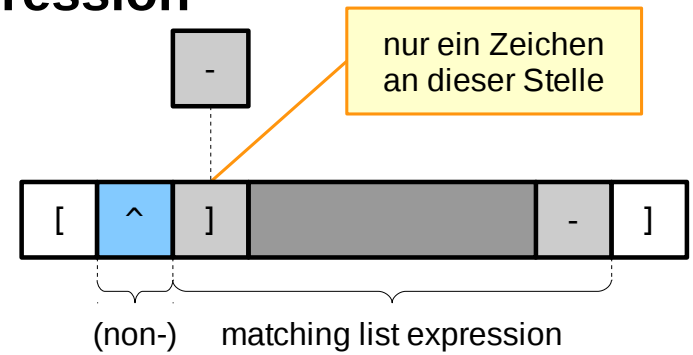
spezielle Zeichen

\	Escape
.	beliebiges Zeichen
[…]	Zeichenklasse
*	Wiederholung – beliebig oft
^	Anker – Zeilenanfang
$	Anker – Zeilenende

spezielle gewöhnliche Zeichen

\(...\)	Gruppierung
\{...\}	Wiederholung – n bist m Mal
n	Rückwärts Referenz

Bracket Expression



spezielle Zeichen

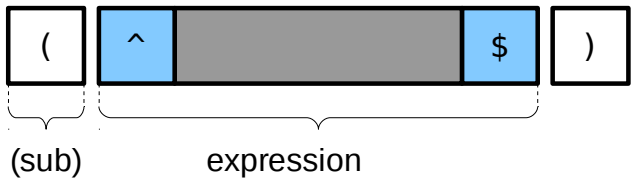
^	keines der Zeichen
…–…	Bereich von … bis …
[:…:]	POSIX Zeichenklasse
[=…=]	Equivalence-Class
[.….]	Collating-Element

POSIX Zeichenklassen

[:alnum:]	[:cntrl:]	[:lower:]	[:space:]	[:alpha:]	[:digit:]
[:print:]	[:upper:]	[:blank:]	[:graph:]	[:punct:]	[:xdigit:]

3.2 POSIX – Extended Regular Expressions (ERE)

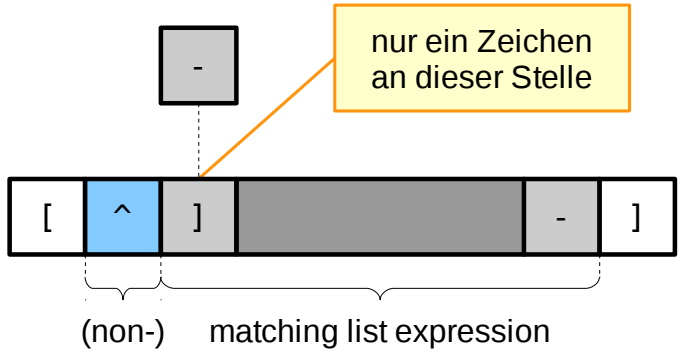
Regular Expression



spezielle Zeichen

\	Escape
.	beliebiges Zeichen
[…]	Zeichenklasse
(...)	Gruppierung
|	Alternative
^	Anker – Zeilenanfang
$	Anker – Zeilenende
?	Wiederholung – maximal einnmal
*	Wiederholung – beliebig oft
+	Wiederholung – mindestens einmal
{...}	Wiederholung – n bist m Mal

Bracket Expression



spezielle Zeichen

^	keines der Zeichen
…–…	Bereich von … bis …
[:…:]	POSIX Zeichenklasse
[=…=]	Equivalence-Class
[.….]	Collating-Element

POSIX Zeichenklassen

[:alnum:]	[:cntrl:]	[:lower:]	[:space:]	[:alpha:]	[:digit:]
[:print:]	[:upper:]	[:blank:]	[:graph:]	[:punct:]	[:xdigit:]

3.3 Perl Compatible Regular Expressions (PCRE)

Kurzübersicht (unvollständig!)

spezielle Zeichen

\	Escape
.	beliebiges Zeichen
[…]	Zeichenklasse
(…)	Gruppierung – mit Speicherung Teilergebnis
(?:…)	Gruppierung – ohne Speicherung
|	Alternative
^	Anker – Zeilenanfang
$	Anker – Zeilenende

Wiederholungen (nicht gierig mit nachgestelltem ?)

?	Wiederholung – maximal einnmal
*	Wiederholung – beliebig oft
+	Wiederholung – mindestens einmal
{…}	Wiederholung – n bist m Mal

Zusicherungen

(?=…)	(?!…)	Nachfolgemuster
(?<=…)	(<!=…)	Vorgängermuster

Optionen (?…)

i	ignore case
U	default ungreedy

spezielle gewöhnliche Zeichen

\w	\W	Wort-Zeichen	(kein …)
\s	\S	Whitespace	(kein …)
\d	\D	Ziffer	(keine …)
\Pc	\pc	Eigenschaft	(keine …)
\b	\B	Anker – Wortanfang	(kein …)
\A		Anker – Stringanfang	
\Z		Anker – String-/Zeilenende	
\z		Anker – (nur) Stringende	
\z		Anker – letzte Treffer-Position	
\n		Rückwärts-Referenz	

Bracket Expression

\	Escape
^	keines der Zeichen
…–…	Bereich von … bis …
[:…:]	POSIX Zeichenklasse