Reguläre Sprachen, Endliche Automaten, und Reguläre Ausdrücke

Programmieren und Software-Engineering
Homomorphismen, Formale Sprachen und Syntax-Analyse

22. Februar 2023

Reguläre Ausdrücke: Theorie

Regulärer Ausdruck über einer Menge Σ von Zeichen

1. jedes Zeichen $z \in \Sigma$ ist ein regulärer Ausdruck
2. sind r und s reguläre Ausdrücke, gilt
   - *Aneinanderreihung:* $r\ s$ ist ein regulärer Ausdruck
   - *Alternative:* $r \mid s$ ist ein regulärer Ausdruck:
   - *Wiederholung:* r^* ist ein regulärer Ausdruck:
   - *Klammerung:* (r) ist ein regulärer Ausdruck:

Beispiel: $\Sigma = \{a,b\}$

- a, b, aa, ab, ba, abba, abbbaaba
- a|b, abb|ba
- a^*, $(a|b)^*$
- $a(a|b)^*b$
- $(a|b)^*abb$
- a^*b^*
- aa^*bb^*
- $a^*(a|b)+$
- $(aa)^*(bb)^*b$
- $(aa|b)^*(a|bb)$

Praxis: **B**asic **R**egular **E**xpressions Standard

`c`	dass Zeichen c	BRE
`c1c2...`	Zeichenfolge $c_1 c_2 \ldots$	BRE
`.`	ein beliebiges Zeichen	BRE
`\`	*escape character*: das folgende Zeichen ist *kein* Steuerzeichen	BRE
`\.`	Punkt-Zeichen	BRE
`[c1c2...]`	eines der Zeichen $c_1, c_2 \ldots$	BRE
`[c1-c2]`	eines der Zeichen zwischen c_1 und c_2 (inkl.)	BRE
`[^c1c2...]`	jedes andere Zeichen als $c_1, c_2 \ldots$	BRE
`[^c1-c2]`	jedes andere Zeichen als eines zwischen c_1 und c_2	BRE
`\w`	Buchstabe, Ziffer, _ : äquivalent `[A-Za-z0-9_]` `\W`: *nicht* ...	BRE
`\s`	Whitespace `\S`: *nicht* ...	BRE
`^`	Zeilenbeginn	BRE
`$`	Zeilenende	BRE
`\b`	Wortgrenze: Wechsel `\w` $\leftrightarrow$ `\W` `\B`: *nicht* ...	BRE
`*`	**Wiederholung** beliebig oft inkl. 0-mal *(lazy: `*?` PCRE)*	BRE

Praxis: **E**xtended und **P**erl **C**ompatible **R**egular **E**xpressions

`+`	Wiederholung, beliebig oft, mindestens 1-mal *(lazy: `+?` PCRE)*	ERE
`{n}`	Wiederholung genau *n*-mal	ERE
`{m,n}`	Wiederholung *m*- bis *n*-mal	ERE
`{m,}`	mindestens *m*-mal	ERE
`{,n}`	höchstens *n*-mal	ERE
`?`	optional (0-mal oder 1-mal)	ERE
`|`	oder (Alternative)	ERE
`\d`	Ziffer: äquivalent `[0-9]` `\D`: *nicht* ...	PCRE
`( ... )`	**Gruppierung**[1] und *back references* `\1`, `\2`, ...	ERE

[1] hauptsächlich für Ersetzungen

Beispiele Zahlendarstellung

Ganze Zahl: 1, 123, +123, -123, 100, 001, 0, +0, -0

```
[-+]?[0-9]+
```

ohne führende Nullen, inkl. Null: 1, 123, +123, -123, 100, 0

```
[-+]?[1-9][0-9]*|0
```

Kommazahl: 3.14, +3.14, -3.14, 3.

```
[-+]?[0-9]+\.[0-9]*
```

ohne Vorkommateil: .14159

```
[-+]?([0-9]+\.[0-9]*|\.[0-9]+)
```

mit Exponent: 3.14E12, 3.14e12, 3.14e-12, 3.14e+12

```
[-+]?([0-9]+\.[0-9]*|\.[0-9]+)([eE][-+]?[0-9]+)?
```

Beispiel: Aufbau einer Textzeile

z. B. `Meier,␣Paul␣F.:␣␣DBI2␣3␣14:00-16:30␣C3.15␣`

1. Zeilenbeginn
2. Familienname (Buchstaben, 1. groß)
3. Beistrich
4. kein oder mehrere Leerzeichen
5. Vorname (Buchstaben, 1. groß)
6. ein oder mehrere Leerzeichen
7. 2. Vorname Abkürzung (Großbuchstabe, Punkt)
8. Doppelpunkt
9. kein oder mehrere Leerzeichen
10. Kürzel (3 Großbuchstaben, Ziffer zwischen 1 und 4)
11. ein Leerzeichen
12. Wochentag als Nummer (1-5)
13. ein Leerzeichen
14. Uhrzeit von (als 2 Ziffern, Doppelpunkt, 2 Ziffern)
15. Bindestrich
16. Uhrzeit bis
17. ein oder mehrere Leerzeichen
18. Raum (Buchstabe A-D, Ziffer 1-5, Punkt, Ziffer, Ziffer 1-9)
19. kein oder mehrere Leerzeichen
20. Zeilenende

Beispiel: Aufbau einer Textzeile (2)

z. B. `Meier, Paul F.:  DBI2 3 14:00-16:30 C3.15 `

Zeilenbeginn	`^`
Familienname (Buchstaben, 1. groß)	`[A-Z][a-z]+`
Beistrich	`,`
kein oder mehrere Leerzeichen	`␣*`
Vorname (Buchstaben, 1. groß)	`[A-Z][a-z]+`
ein oder mehrere Leerzeichen	`␣+`
2. Vorname Abkürzung (Großbuchstabe, Punkt)	`[A-Z]\.`
Doppelpunkt	`:`
kein oder mehrere Leerzeichen	`␣*`
Kürzel (3 Großbuchstaben, Ziffer zwischen 1 und 4)	`[A-Z]{3}[1-4]`
ein Leerzeichen	`␣`
Wochentag (1-5)	`[1-5]`
ein Leerzeichen	`␣`
Uhrzeit von (als 2 Ziffern, Doppelpunkt, 2 Ziffern)	`[0-9]{2}:[0-9]{2}`
Bindestrich	`-`
Uhrzeit bis	`[0-9]{2}:[0-9]{2}`
ein oder mehrere Leerzeichen	`␣+`
Raum (Buchstabe A-D, Ziffer 1-5, Punkt, Ziffer, Ziffer 1-9)	`[A-D][1-5]\.[0-9][1-9]`
kein oder mehrere Leerzeichen	`␣*`
Zeilenende	`$`

Beispiel: Aufbau einer Textzeile (3)

z. B. `Meier, Paul F.:  DBI2 3 14:00-16:30 C3.15 `

Regulärer Ausdruck für die ganze Zeile

```
^[A-Z][a-z]+, *[A-Z][a-z]+ +[A-Z]\.:
 *[A-Z]{3}[1-4] [1-5] [0-9]{2}:[0-9]{2}-[0-9]{2}:[0-9]{2} +
[A-D][1-5]\.[0-9][1-9] *$
```

Mockaroo: Generierung statt Match, d. h. * ⇒ {0,9}, + ⇒ {1,9}

```
[:upper:][:lower:]{1,9}, {0,9}[:upper:][:lower:]{1,9} {1,9}[:upper:].:
 {0,9}[:upper:]{3}(1|2|3|4) (1|2|3|4|5) \d{2}:\d{2}-\d{2}:\d{2} {1,9}
(A|B|C|D)(1|2|3|4|5).\d(1|2|3|4|5|6|7|8|9) {0,9}
```

Zeichenklassen (BRE)

`[:upper:]`: Großbuchstaben, `[:lower:]`: Kleinbuchstaben, `[:alpha:]`: Buchstaben, `[:digit:]`: Ziffern, `[:alnum:]`: Buchstaben und Ziffern, `[:blank:]`: Leerzeichen oder Tabulator, `[:punct:]`: Sonderzeichen, ...

Literaturübersicht I

[1] **Berger, Krieger, Mahr: "Grundlagen der elektronischen Datenverarbeitung", Skriptum**

[2] Dirk W. Hoffmann: "Theoretische Informatik", Hanser, 3. Auflage

[3] Gernot Salzer: "Einführung in die Theorie der Informatik", Skriptum, TU Wien, 2001

[4] Wikipedia (Englisch): `https://en.wikipedia.org/`

[5] Wikipedia (Deutsch): `https://de.wikipedia.org/`