Bundeswettbewerb Informatik
Runde 1

1 Störung

Die Lösung dieses Problems liegt bereits in der Natur des Problems. Es gibt einen endlich langen Eingabestring, der somit aus einem endlichen Eingabealphabet Σ besteht. Diesen gilt es in einer endlichen Folge von Schritte auf Validität zu überprüfen. Somit lässt sich das gewünschte Verhalten über einen endlichen Automaten erreichen. Dieser endliche Automat besteht neben dem Eingabealphabet Σ aus einer endlichen Menge an Zuständen Q, von denen einer ein definierter Startzustand $Q_0 \in Q$ und eine Teilmenge die definierten Endzustände $Q_A \subseteq Q$ ist. Der Automat wechselt zwischen seinen Zuständen Q basierend auf seiner Übergangsfunktion δ.
Diese Art von Automaten können in einem Zustandsdiagramm abgebildet. Nachstehend ist ein Automat abgebildet, der seine Eingabe auf die Zeichenfolge `'Ha'`, gefolgt von beliebig vielen `'a'`, gefolgt von `'ll'`, gefolgt von beliebeig vielen `'o'` überprüft.

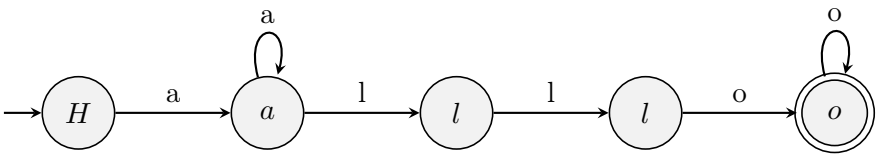

Abbildung 1: Beispielautomat - `'Haaaaallooo'`

Ein regulärer Ausdruck (Regular Expression oder Regex) ist eine Konvention zur Darstellung dieser Automaten in kurzen, prägnanten Zeichenketten. Der obenstehende Automat kann so als `'Ha+llo+'` ausgedrückt werden. Hierbei kodiert das `'+'` als Kontrollzeichen für das ein oder mehrfache Vorkommen des vorhergegangenen Zeichens.
Mithilfe dieser Kontrollzeichen kann der Ausdruck `[A-Za-z',.!?ßäöü»«]+` gefunden werden, der für ein Wort beliebiger Länge, mit 0 oder mehr der erwähnten Sonderzeichen kodiert.

Die in einer gestörten Nachricht, durch Unterstriche markierten, fehlenden Wörter können dann durch diesen Ausdruck ersetzt werden. So erhält man für `stoerung1.txt` den Audruck
`'ich muß [A-Za-z',.!?ßäöü»«]+ clara [A-Za-z',.!?ßäöü»«]+`.

Aus diesem generiert das nativ bereitgestelle Python Modul *re* dann einen der beschrieben Automaten. Mit der Methode `re.findall(expression, word)` einen der oben beschrieben Automaten und durchsucht mit ihm die Eingabe `word`. Zurückgegeben wird eine Liste aller Treffer. In diesem Falle:
`['ich muß in clara verwandelt', 'ich muß doch clara sein,']`
Analog zu diesem Verfahren ergeben sich folgende Ausgaben für die restlichen Testfälle:

Eingabe	Ergebnis
stoerung0.txt	['das kommt mir gar nicht richtig vor']
stoerung1.txt	['ich muß in clara verwandelt', 'ich muß doch clara sein,']
stoerung2.txt	['fressen katzen gern spatzen?', 'fressen spatzen gern katzen?«']
stoerung3.txt	['das spiel fing an.', 'das publikum fing an,']
stoerung4.txt	['ein sehr schöner tag']
stoerung5.txt	['wollen sie so gut sein']