

20 February 2025

Thursday

Announcement

Assignment Submission week 3 Quiz 2025

regex identifier ৩৩
৩৩৩, constant ৩৩
৩৩৩ lab ৩

Common RegEx Symbols

Kleene Closure

$a^* \rightarrow \epsilon, a, aa, aaa, \dots$

Concatenation

$a \cdot b = ab$

Positive Closure

$a^+ \rightarrow a, aa, aaa, \dots$

minimum একবার হলেও থাকবে

Union

$a \cup b$ বা $a|b$ দুটি এখন
কিছু হয়, (a or b)

$(a|b)^* \rightarrow \epsilon, a, b, abba, \dots$

Bracket এর precedence সবার থেকে বেশি

এরপর সবচেয়ে বেশি higher precedence সবার Kleene Closure

Positive closure

a^*b+c

Kleene Closure, Positive Closure

Concatenation

Union

precedence বাড়ছে

$[]$

\rightarrow exactly ekta character at a time থাকবে

example,

$[brn]ed \rightarrow bed, red, ned$

$[brn]^*$

\rightarrow Kleene Closure সাক্ষাৎ এতে choose portion multiple time হবে

$[-]$

range বুঝায়

$[a-z]^*$

\rightarrow zacb.....

a থেকে z range এর মধ্যে যে কোন character choose হবে তার ক্ষেত্রে Kleene Closure জায়গা so choosing multiple times

$a^* \rightarrow \epsilon, a, aa, aaa, aaaa, \dots$

$a^+ \rightarrow a, aa, aaa, aaaa, \dots$

Now suppose you have aa exactly match string aa . So you can say that aa is a string which matches the string aa . So you can say that aa is a string which matches the string aa .

$a\{m\} \rightarrow a$ will be repeated m times. (fixed)

$\therefore a\{2\} = aa$ fixed 2 times.

at least m times repeat a m times.

$a\{m, \}$ \rightarrow at least m times.

example: $a\{2, \}$ $\rightarrow aa, aaa, aaaa, \dots$

at least m times and at most n times repeat a .

$a\{m, n\} \rightarrow$ at least m times at max n times.

example: $a\{2, 4\} \rightarrow aa, aaa, aaaa$

"a" \rightarrow exactly a character.

• normally - sign দিয়ে range বুঝায়,

■ Suppose তলি এম string "a-z" regEx দিয়ে express করতে চাই, a to z তা easily express করতে পারি; but hyphen (hyphen) express করা যায়?

→ "-" এটার literally hyphen বুঝায়

→ - এটার দিলে range বুঝায়,

→ \-

literal character { "-"
"\-"

→ এতে ২ ডাফার আছে।
কোনটা একটা
express করতে
হবে,

∴ "a-z" can be expressed as,

a."-".z or a.\-\.z

compiler এ defined জিনিসগুলো মত - , . , " " string এ মত literal character মতটা express করে উভা express করা হবে.

• \$ দিয়ে end of line বুঝায়,

ab \$ cd → ab
cd

regEx
↓
NFA
↓
DFA
then
match
করে.

• ? দিয়ে বুঝায় Zero or one occurrence.

colou?r → colon, colour.

bracket এ মত মত
মত মত Question মত,
মত মত মত মত
মত মত,

Theory to regEx generate
করে তবে then ডি-গেন
DFA

■ Create 5 strings using the given regex,

$[a-zA-Z-] [a-zA-Z-]^*$

- ⇒
- a
 - zaAZb
 - Back-
 - bE-n
 - apple
 - - Ant

Regular Definitions,

- regex টের মাত ব্যাবহার না কৈখ নাহা নো তাকরা তকৈ এটা definition এ মতৈ কৈখ দিত গরি, "-"
- kind of like a variable ; এ মতৈ expression store কৈত জাখত গরি

letter → $[a-zA-Z-]$ letter definition এ corresponding action.

$[a-zA-Z-] [a-zA-Z-]^* \rightarrow [letter] [letter]^*$

digit → $[0-9]$

integer → $[digit] [digit]^*$

$[a,b,c]$

0-9

(a,b,c)

$[0-9]$

(0-9)

cd (a* b c) e*

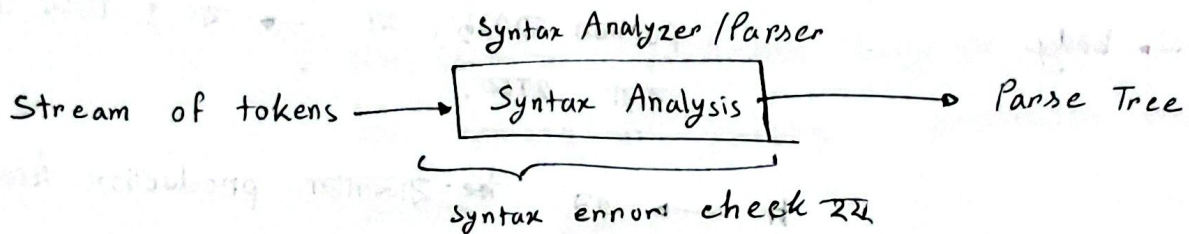
$(a|b)^*$ → ক্ষেত্র b এত কৈত Kleene closure.

(alb)*

- Quotation match string এ চাইলে "" বা \
- lexeme match করতে regex এর মাধ্যমে then token generation.

Syntax Analysis Phase

- Syntax analysis phase এ parse tree generate হয়,
- বর্তমানে parse tree এর পরিবর্তে stack দিয়ে parsing হয়,
- Syntax error checking হয় CFG দিয়ে,



- CFG এর মাধ্যমে match করতে, automate করতে, generate parse tree.
- CFG থেকে টার্মিনাল parse tree (table) generate হয়, যেটা টার্মিনাল থেকে check হয়
- Parser, parsing করতে generate parse table এর মাধ্যমে টার্মিনাল code টার্মিনাল চিহ্ন তাকান চিহ্ন- check করতে,
- Parser always left to right parse করতে. এর জন্য token generation 3 left to right হয়.
- Parse tree সঠিক নয় means my stream of tokens are correct.
- CFG → Context Free Grammar

- CFG এ 4 টি term থাকে,

- (i) Terminal *
- (ii) Non Terminal *
- (iii) Start *
- (iv) Production Rules *

$$\left. \begin{array}{l} S \rightarrow E \\ E \rightarrow T + E \\ T \rightarrow F \\ F \rightarrow id \end{array} \right\} \text{একটি grammar}$$

→ চিহ্ন বা : শব্দে পাঠ্য = শব্দে পাঠ্য

- $Start \rightarrow expression$
- $expression \rightarrow term + factor$
- $term \rightarrow factor$
- $factor \rightarrow id$

একটি CFG

এখানে, ৭টি production rule আছে, এদের production rule এ ২টি-

head \rightarrow body

portion আছে. যা \rightarrow বা : দিয়ে শুরু বা =

\rightarrow এর ~~কি~~ ক্রমাগত production head

\rightarrow এর ক্রমাগত production body

Production Rules: Production Head: Production Body

Start:

- Start দিয়ে starting point দিতে হবে, অন্য grammar হতে উদ্ভূত হলে start হতে হবে,
- Starting symbol বা শব্দটি উল্লিখিত নাও থাকতে পারে not necessary, মাঝখানে থাকতে পারে,

Start: Start

- Start এর জন্য ২,৩ টি starting point ও থাকতে পারে, উদ্ভূত হলে উল্লিখিত করে দিতে হবে, Start এর জন্য উল্লিখিত production rules ও থাকতে পারে,

Terminal : Direct token ଯାହାକି lexer ଦେଇ pass ହେଉଛି,

Non Terminal: • Grammar portion

- ଯାହା production head ଏ ଯାହା, production head ଏ ଚାଲିଥାଏ ନା ଚାଲିଥାଏ ଦିଅନ୍ତୁ, non terminal production body ଠାରେ ଯାହାକି production head ଏବଂ ଯାହାକି ଯାହା,
- Token ନା.
- ଚାଲିଥାଏ ନା ଚାଲିଥାଏ production head ଏ ଦିଅନ୍ତୁ.
- Non terminal production body ଠାରେ use କରନ୍ତୁ ତା ଚାଲିଥାଏ ନା ଚାଲିଥାଏ production head ଏ ଯାହାକି ହେଉ,
- Non terminal here:
expression, term, factor

id, +, * ହେଉ direct tokens ଯାହା ଏହାକି terminal .

Terminal: id, +, *

↳ Production body ଠାରେ ଯାହା, ଯାହାକି production head ଏ ଯାହା ନା,

"Start one kind of non terminal, special kind of non terminal, as it denotes the starting point."