COMP 408 Advanced Topics in Artificial Intelligence

Lecture 1

Introduction and Regular Expressions

8 / 2 / 2025

Some Slides are by D. Jurafsky and J. M. Martin

Course Information

- Course Code: COMP 408
- Course Title: Advanced Topics in Artificial Intelligence:
  - Part 1: Natural Language Processing (NLP)
  - Part 2: Multi Agent System (MAS)   Dr. Hewayda
- Number of Credit Hours: 3 (3 hours Lecture + 0 Lab.)
- Prerequisite : -
- Final Exam (duration): 3 hours
- Total Marks: 150
  - 105 Final Exam.
  - 37 Mid-Term Exam.
  - 8 Oral Exam.

Part 1: NLP

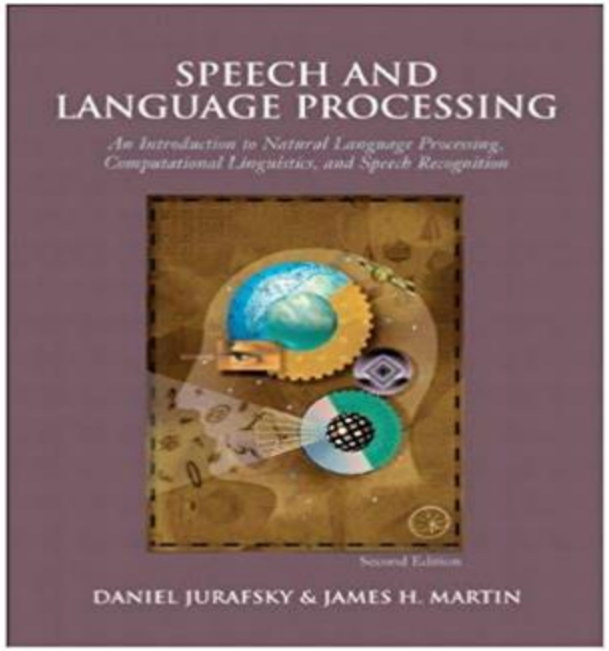
Objectives

- Learn the broad topics, fundamental concepts, and some common techniques in the field of natural language processing.
- Understand the capabilities, limitations, and promise of NLP.
- Understanding of the computational properties of natural languages and the commonly used algorithms for processing linguistic information.

Textbook

Daniel Jurafsky and James H. Martin (2025), Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Model , 3rd edition, Pearson Education Limited.

https://web.stanford.edu/~jurafsky/slp3/



Natural Languages

- One of the fundamental aspects of human behavior and is a crucial component of our lives.
- A communication mechanism whose medium is text or speech.

Natural Language Processing (NLP)

- A field of computer science and artificial intelligence that focused on the technology and algorithms of processing natural languages.

Why processing natural languages?

- There are two main reasons to process natural languages:
  - Enable machines to communicate with humans.
    - Alan Turing proposed his Test based it on language.
  - Acquire information from written language.
    - Text summarization.
    - Sentiment analysis.
    - Search for relevant text document.
    - Classify text documents.
    - Topic modeling; identify the topic of a given text.

Common Applications of NLP

- Spell Checking
- Sentiment Analysis
- Information Retrieval
- Machine Translation
- Question Answering
- Email Classification

Categories of Knowledge

Types of knowledge required to deal with natural language:

- Phonology
- Morphology
- Syntax
- Semantics
- Pragmatics

Phonetics

- Study of the speech sounds that make up languages.

Morphology

- Word formation and their meaning.
- How words can be constructed from more basic meaning units called morphemes:
- A morpheme is the primitive unit of meaning in a language:
  - The meaning of the word "friendly" is derivable from the meaning of the noun "friend" and the suffix "-ly", which transforms a noun into an adjective.

English morphology

- English affixes can be divided to:

  prefixes, suffixes

- For example:
  - Eats consists of the stem eat and the suffix -s
  - Foxes consists of the stem fox and the suffix -es
  - Unstable consists of the stem stable and the prefix un-

Arabic morphology

- Arabic affixes can be divided to:

  prefixes, suffixes, infix

- For example:
  - كتبوا consists of the stem كتب and the suffix وا
  - يكتب consists of the stem كتب and the prefix ي

Syntax

- How words can be put together to form grammatical sentences:
  - I saw a woman in the garden (grammatical)
  - * I a woman saw in the garden (ungrammatical)
- The statement “I saw a man with a telescope” is grammatical; has two parse trees each of the trees corresponds to different meaning.
  - This is called syntactic ambiguity.

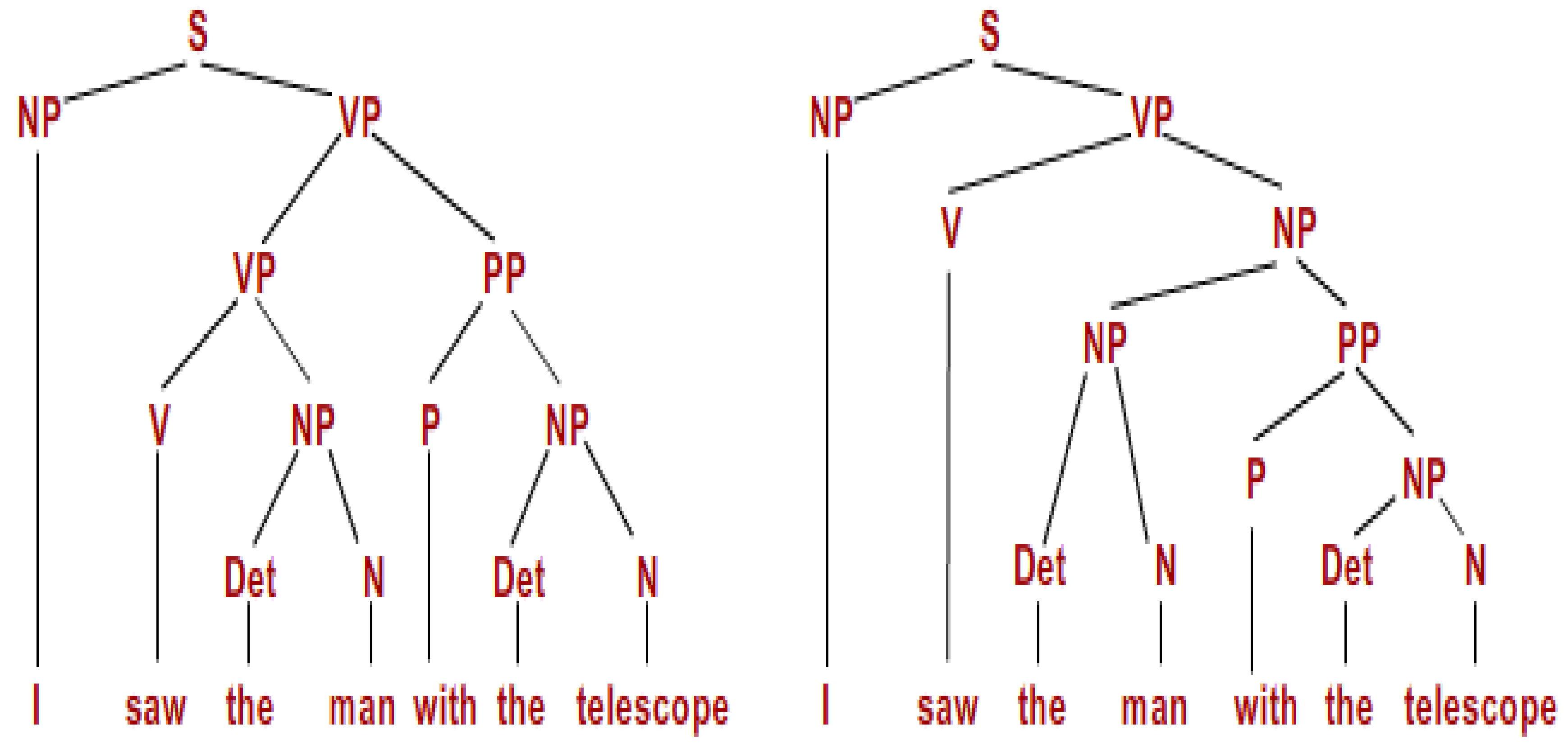
Syntax Cont.

- A sentence is grammatical if it has a parse tree or derivation.
- Consider the context-free grammar:

S → NP VP	NP → I
NP → NP PP	P → with
NP → Det N	Det → the
VP → VP PP	N → man
VP → V NP	N → telescope
PP → P NP	V → saw

Syntax Cont.

Two parse trees can be constructed, for the sentence, from the pervious grammar



Semantics

- Study of the word meaning and how the meanings combine in sentences to form sentence meanings.
- This is the study of context independent meaning - the meaning a sentence has regardless of the context in which it is used:
  - The word bank has two different senses (meaning)
    - Ambiguity in the sense of the word

Pragmatics

- Study of the meaning of languages in contexts.

Question

"The tires are brand new."

- Given that the person uttering this sentence is responding to a complaint that the car is too cold, this sentence is:
  A. Syntactically incorrect.
  B. Semantically incorrect.
  C. Pragmatically correct.
  D. Syntactically and semantically correct.

Question

"The tires are brand new."

- Given that the person uttering this sentence is responding to a complaint that the car is too cold, this sentence is:
  A. Syntactically incorrect .
  B. Semantically incorrect.
  C. Pragmatically correct.
  D. Syntactically and semantically correct.

What makes NLP hard?

- **Ambiguity:** Human language is full of ambiguities.
  - Words can have multiple meanings depending on context, which makes it hard for machines to understand the intended meaning.
- **Context:** Grasping the context is crucial.
  - Words and sentences are often interpreted based on the surrounding text, previous sentences, or even real-world knowledge.
- **Diversity of Languages**: Each language has its own set of grammar rules, syntax, idioms, and slang.
  - NLP models need to handle this diversity to understand and generate text accurately.

What makes NLP hard? Cont.

- **Idiomatic Expressions**: Phrases like "فات القطار" or "piece of cake" can be challenging since their meanings are not literal.
- **Sarcasm and Irony**: Detecting sarcasm or irony is tough because it relies heavily on tone, context, and cultural nuances, which are difficult for machines to grasp.
- **Named Entity Recognition (NER):** Identifying proper nouns, like names of people, places, or organizations, requires understanding of the text and sometimes external knowledge.
- **Morphology and Syntax**: Human languages have complex morphology (word forms) and syntax (sentence structures) that need to be parsed accurately.

Ambiguity Example

- Find at least 5 meanings of this sentence:
  - I made her duck

Ambiguity Example

- Find at least 5 meanings of this sentence:
  - I made her duck

1. I cooked waterfowl for her benefit (to eat)
2. I cooked waterfowl belonging to her
3. I created the (plaster?) duck she owns
4. I caused her to quickly lower her head or body
5. I waved my magic wand and turned her into undifferentiated waterfowl

Ambiguity Example

- I caused her to quickly lower her head or body
  - **Lexical category**: "duck" can be a Noun or a Verb
- I cooked waterfowl belonging to her.
  - **Lexical category:** "her" can be a possessive ("of her") or dative ("for her") pronoun
- I made the (plaster) duck statue she owns
  - **Lexical Semantics:** "make" can mean "create" or "cook"

Ambiguity

- Grammar: Make can be:
  - **Transitive: (verb has a noun direct object)**
    - I cooked [waterfowl belonging to her]
  - **Ditransitive: (verb has 2 noun objects)**
    - I made [her] (into) [undifferentiated waterfowl]
  - **Action-transitive (verb has a direct object and another verb)**
    - I caused [her] [to move her body]

Ambiguity

- **Phonetics!**
  - I mate or duck
  - I’m eight or duck
  - Eye maid; her duck
  - Aye mate, her duck
  - I maid her duck
  - I’m aid her duck
  - I mate her duck
  - I’m ate her duck
  - I’m ate or duck
  - I mate or duck

Example 1

I sent her messages

Has two different meanings

a. I sent some messages to her

b. I sent the messages written by her

- There is ambiguity in the word *her*, since it can be a dative pronoun or a possessive pronoun.
- There is also a syntactic ambiguity because the verb send is syntactically ambiguous, it may be transitive with one object as in (b) or with two objects as in (a).

Example 2

He drew one card

Has two different meanings

a. He created one card having a picture.
b. He chose one card from a set of cards.

- The different meanings are because of semantic ambiguity because the verb draw has many senses (meaning) such as created a picture and choose.

Regular Expression

Chapter 2 of the textbook

A tool for describing text patterns

Regular Expressions and Text Searching

- Regular expressions are a compact textual representation of a set of strings representing a language
  - In the simplest case, regular expressions describe regular languages
    - Here, a formal language means a set of strings from a given some alphabet.

Regular expressions

- A formal language for specifying text strings
- How can we search for mentions of these cute animals in text?
  - woodchuck
  - woodchucks
  - Woodchuck
  - Woodchucks
  - Groundhog
  - groundhogs

Regular Expressions: Disjunctions

- Letters inside square brackets [] specify disjunction of characters
- [wW] matches w or W

Pattern	Matches	Example
`/[wW]oodchuck/`	Woodchuck, woodchuck	Woodchuck
`/[abc]/`	‘a’, ‘b’, or ‘c’	I like Java
`/[1234567890]/`	Any digit from 0 to 9	7 or 5

Regular Expressions: Disjunctions

- Ranges [A-Z]

Pattern	Matches	Example
/[A-Z]/	An upper case letter	Drenched Blossoms
/[a-z]/	A lower-case letter	my beans were impatient
/[0-9]/	A single digit	Chapter 1: Down the Rabbit Hole
/[2-4]/	2, 3, or 4	Chapter 2

Question

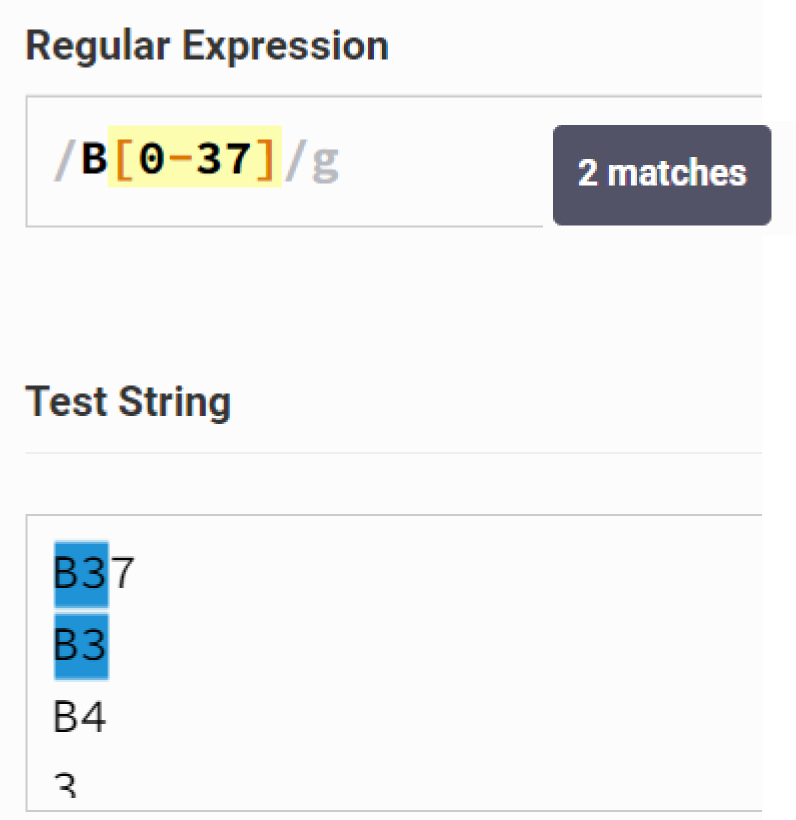
- The regular expression B[0-37] matches:
    1. B37
    2. B3
    3. B4
    4. 3

Question

- The regular expression B[0-37] matches:

(B[0, 1, 2, 3, 7])

1. B37
2. B3
3. B4
4. 3



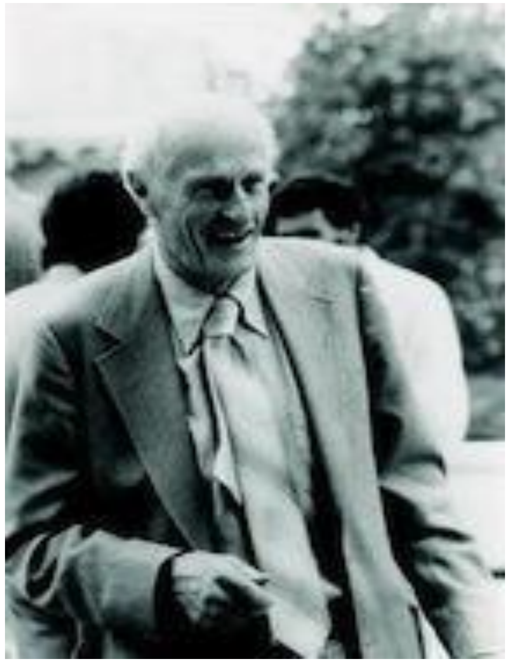
Regular Expression: More Disjunction

- The pipeline symbol ‘|’
- Examples,
  - /a|b|c/ = /[abc]/
  - /[gG]roundhog/ | /[Ww]oodchuck/

Regular expressions: ? * + .

? Marks optionality of the previous expression.
* Marks zero or more occurrences of the previous expression.
+ Marks one or more occurrences of the previous expression.
Period (.) Marches any single character.

Pattern	Matches	Example
/Colou?r/	previous character optional	Color Colour
/oo*h!/	Zero or more previous character	oh! ooh! oooh! ...
/[ab]*/	Zero or more a's or b's	aaa, abab, bbb
to*	0 or more of previous char	t to too tooo
to+	1 or more of previous char	to too tooo toooo
/beg.n/		beg'n begin begun



Stephen C Kleene

Kleene *, Kleene +

Note

- The period (.) is often used with Kleene star (*), to mean any string of characters:
    - The regular expression /cat.*cat/ matches any line with word cat at the beginning of the line and then any sequence of characters then the word cat.

Example

- How can we search for any of these?
  - woodchuck
  - woodchucks
  - Woodchuck
  - Woodchucks
  - Groundhog
  - groundhogs

Example

- How can we search for any of these?
  - woodchuck
  - woodchucks
  - Woodchuck
  - Woodchucks
  - Groundhogs
  - groundhogs

/[wW]oodchucks?| [Gg]roundhogs?/

A word begins with either w or W and may ends with s or a word begins with G or g and may end with s

Regular expressions : Negation in Disjunction

Carat (ˆ) as first character in [] negates the list

Note: Carat means negation only when it's first in []

Pattern	Matches	Examples
/[ˆA-Z]/	Not an upper-case letter	Oyfn pripetchik
/[ˆSs]/	Neither 'S' nor 's'	I have no exquisite reason"
/[eˆ]/	either 'e' or '^'	Look here
/aˆb/	The pattern a caret b	Look up a^b now

Regular expressions: anchors ˆ, $, \b, and \B

- The ^ matches the start of the line (if it appears at the beginning of the regular expression.)
- The $ matches the end of the line.
- \b matches word boundary and \B matches non-word boundary

Pattern	Example
/^The/	The dog
/^[^A-Za-z]/	1 "Hello", the first ^ matches the start of line, the ^ inside [] means not.
/cat$/	The big cat.

Note

- The caret, ˆ, has three uses:
  1. To match the start of the line
     - /ˆThe/ matches the word The at the start of the line
  2. To indicate negation inside of square brackets
     - /[ˆA-Z]/ matches non-uppercase letter
  3. To mean a caret
     - /[abˆd]/ matches a, b, ˆ, or d
     - /aˆb/ matches aˆb

Regular expressions: { }

Pattern	Match	Example
/the{3}/	3 occurrences of the last letter e	theee
/the{2, 4}/	From 2 to 4 occurrences of the last letter e	thee theee theeee
/the{2,}/	At least 2 occurrence of the last letter e	thee, theee, …

RE operator precedence

- Parenthesis
- Counters
- Sequences and anchors
- Disjunction

( )

*   +   ? {}

the ^my end$

|, []

Higher

Lower

Notes

1. Since counters have a higher precedence than sequence of characters:
   - the RE /the*/ matches theeee not thethe.
2. Since sequence have a higher precedence than disjunction:
   - The RE /the|any/ matches the or any not theny.

More operators

RE	Match
*	an asterisk “*”
\.	a period “.”
\?	A question mark
\d	[0-9], any digit
\D	[^0-9], any non-digit
\w	[a-zA-Z0-9_], any alphanumeric or underscore
\W	[^\w], a non-alphanumeric e.g., ? ; + * -
\s	[\r\t\n\f], white space
\S	[^\s], non-whitespace

Example 1

- Write a regular expression to match the word "the" (or "The") in a text.
  - /the/ misses capitalized T (The)
  - /[tT]he/ incorrectly matches others or theory
  - `/\W[tT]he\W/`

False positives and false negatives

- The process we just went through was based on two fixing kinds of errors

1. Not matching things that we should have matched (The)

   False negatives

2. Matching strings that we should not have matched (there, then, other)

   False positives

Characterizing work on NLP

In NLP we are always dealing with these kinds of errors.

Reducing the error rate for an application often involves two antagonistic efforts:

All

- Increasing coverage (or *recall*) (minimizing false negatives).
- Increasing accuracy (or *precision*) (minimizing false positives)

only

Example 2

- Find regular expression to express the language with the following strings (sheep language):

baa! baaa! baaaa!

…

/baaa*!/ or

/baa+!/

Example 3

- Write a regular expression to match the words cat and dog in a text:
  - / \b cat | dog \b /
  - /[catdog]/ (not correct, why?)
- Write a regular expression to match both guppy and guppies:
  - /gupp (y|ies)/
  - /guppy| ies (not correct, why?)

s|r	matches one string s or r
[ab]	matches one character a or b

Example 4

- A regular expression that matches 0 or more a’s:

  /a*/

- A regular expression that matches at least one a’s:

  /aa*/

Example 5

- A regular expression that matches a single digit
  - /[0-9]/ or
  - \d (any digit)
- A regular expression that matches any non-digit
  - \[^0-9]/ or
  - \D
- A regular expression that matches an integer:
  - /[0-9][0-9]*/ or
  - /[0-9]+/

Example 6

- Give regular expression to represent the sets:
  1. {lass, class, glass}

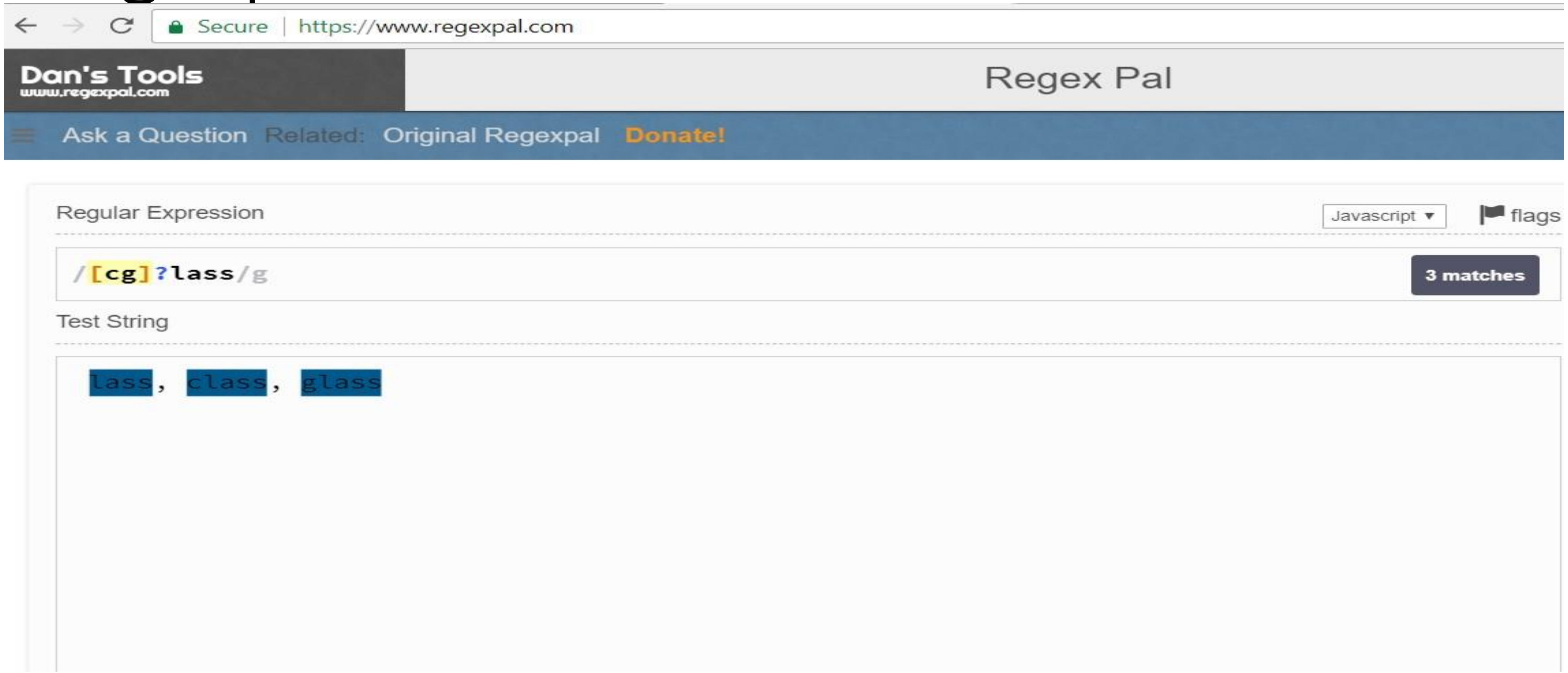
     /\b[cg]?lass\b/   or   /\b(c|g)? Lass\b/

  2. {jet, pet, net}

     /\b[jpn]et\b/   or

     /\b jet | pet | net \b/

regexpal Software



Example 7

- Give regular expression to represent the sets:
  1. {sun, sunday, sunrise, sunset}

  /sun(ε |day |rise |set)/

  or

  /sun(day|rise|set)?/

Question

Select the correct regular expression that describes the set of all lower-case alphabetic strings with letter b as a second letter:

1. /[a-z]*b[a-z]/
2. /[a-z]b[a-z]*/
3. /[a-z]+b[a-z]*/
4. /[a-z]*b[a-z]*/