Décrire et manipuler un document numérique

Cours 3 : Techniques de recherche avancées

Loïc Grobol <lgrobol@parisnanterre.fr>

2024-03-11

Google NGram

Permet de faire des requêtes dans Google Books :

- n-grammes de mots
- n-grammes de catégories grammaticales (parties du discours)

Exemple : « vélocipède » vs « vélo » vs « bicyclette »

→ L'évolution des fréquences relatives des n-grammes dans le corpus

Questions d'usage

Dit-on « autant pour moi » ou « au temps pour moi » ?

« par contre » ou « en revanche »

« *the United States are* » ou « *the United States is* »

- Anglais Américain
- Anglais Britannique

Évolution des nominations

Ouvriers, travailleurs ou salariés

Et au pluriel

Féminisation des noms de métiers

et des fonctions

Le joker *

Le token `*` : remplace n'import quel mot.

Exemple : `une colère *` donne les fréquences de trigrammes dont les deux premiers mots sont « une » et « colère »

Expressions régulières

Expressions régulières = ER = Regex

- Outil de recherche dans des textes
- Permet de trouver des séquences correspondant à un motif (patron). Ex :
  - Même préfixe / suffixe / contexte
  - Dates
  - Adresses
  - URL

Définition de critères de recherche pour des chaînes de caractères

---

Outils :

- Éditeur de texte brut : Notepad++ (Windows), Kate/Gedit (Linux), CotEditor (Mac)…
- Traitement de texte : Google Drive, Word, Writer, Pages…
- Test en ligne : https://regex101.com/
- Langage de programmation : java, perl, python, PHP…

Dans un éditeur de texte

1. Récupérez le fichier `clg.txt`
2. Ouvrez-le avec l'éditeur de votre choix (Clic-droit sur le fichier + ouvrir avec)
3. Ctrl+f pour accéder à la fonction de recherche/remplacement
    - Cochez « expression régulière » (ou `.*`)

Principe général

- Règle générale : par défaut chaque caractère d'une expression correspond (*match*) à ce caractère
    - `madeleine` reconnaît `madeleine` mais pas `Madeleine`
    - `France 2` reconnaît `France 2` mais pas `France2` ni `France 3`
- Cas particuliers : les **métacaractères**
    - Quantificateurs
    - Opérateurs
    - Caractères spéciaux
    - Classes de caractères

Les quantificateurs

Ils portent sur l'élément qui précède et indiquent une répétition

Quantificateur	`?`	`*`	`+`
Signification	« 0 ou 1 fois »	« 0 ou plusieurs fois »	« 1 ou plusieurs fois »

- `jolie?` matche `joli` et `jolie`
- `coup?s` matche `cous` et `coups`, mais pas `coupa`
- `jolie*` matche `joli`, `jolie`, `joliee`, `jolieee`…
- `jolie+` matche `jolie`, `joliee`, `jolieee`…

Exercice : que reconnaissent les expressions suivantes ?

1. `pommes?`
2. `a?typique`

3. `lo*l`
4. `loo*l`
5. `lo+l`

(Tester vos réponses dans un éditeur)

Caractères spéciaux

Caractère	Sens	Exemple
`.`	n'importe quel caractère	`p.re` : pare, pire, père…
`^`	début de ligne	`^Je` (début de ligne)
`$`	fin de ligne	`informatique$` (fin de ligne)
`\b`	frontière de mot (début ou fin de mot)	`\bjour` : `journée` mais pas `ajouré`
`\n`	saut de ligne	`a\n` : `a` puis saut de ligne
`\t`	tabulation	`a\t` : `a` puis tabulation

Exercices : à quoi correspondent ces expressions ?

1. `Nom\tPrénom`
2. `\bcap\b`
3. `\bcap`
4. `premier paragraphe\ndeuxième paragraphe`

Déspécialisation

On a vu un certain nombre de métacaractères (les caractères avec une signification spéciale).

Pour les reconnaître littéralement (les déspécialiser), il faut les faire précéder d'un anti-slash `\` :

- `\^machin` : reconnaît `^machin`
- `3\.0` reconnaît `3.0` mais pas `310`
- Pareil pour `$`, `[`, `]`, `(`, `)`, `*`, `+`, `?`, `{`, `}`

Et pour représenter un anti-slash ? Pareil

- `machin\\bidule` reconnaît `machin\bidule`

Les opérateurs

Alternatives

Le *pipe* indique une alternative :

- `machin|truc` reconnaît `machin` et `truc`.
- `je|tu|nous` reconnaît `je`, `tu` et `nous`.
- `je|tu|[nv]ous` reconnaît `je`, `tu`, `nous` et `vous`.

On les utilise le plus souvent avec des parenthèses

- `(m|r)aison` reconnaît `maison` et `raison`.
- `(il|elle|iel)s?` reconnaît `il`, `elle`, `iel` et leurs versions plurielles.

Ensembles de caractères

Les crochets indiquent un ensemble de caractères à reconnaître :

- `[ab]` reconnaît une lettre : `a` ou `b`
- Avec un tiret : `[a-c]` reconnaît les lettres de `a` à `c`
- `[a-z]` reconnaît les lettres de l'alphabet en minuscules
- `[a-zA-Z]` reconnaît aussi les majuscules*

Exercice :

Que reconnaît :

1. `[ab][ab]`
2. `[MmRr]aison`

Écrire une expression régulière qui reconnaît exclusivement :

1. `moi`, `toi` et `soi` .
2. `ci`, `si`, `ce` et `se`.
3. Les nombres à deux chiffres.

(Tester vos réponses dans un éditeur)

Avec un circonflexe au début : négation de l'ensemble

- [^0-9] reconnaît tout sauf des chiffres
- [^A-Z] reconnaît tout sauf des majuscules sans diacritiques

---

On peut les combiner avec les quantificateurs :

- `[aeiouy]+` reconnaît une suite de voyelles : `a`, `aa`, `ae`, `iey`…
- `[^<]*` reconnaît une suite de caractères différents de `<`

Groupes

Les parenthèses permettent de construire des groupes

- `(ab)+` reconnaît une suite de `ab` : `ab`, `abab`, `ababab`…
- `([ab]c)+` reconnaît `ac`, `bc`, `acac`, `acbc`, `bcac`…
- `([aeiouy][^aeiouy])+` reconnait une suite de couples voyelle-consonne.

Exercice : que reconnaissent les expressions suivantes ?

1. vert(es)?
2. (anti|pro)nucléaire
3. Wiki[a-z]+
4. Wiki[a-z]*

Exercices

Dans le fichier `clg.txt` :

1. Combien trouvez vous de match pour chacun des regex suivantes ? Que signifient-elles ?
    1. Recherchez `^.évolution`.
    2. Recherchez `r?évolution`
2. Combien y a-t-il de mots…

1. de 3 lettres finissant par `r` ?
2. de 4 lettres au plus commençant par `r` ?
3. commençant par `a` et finissant par `er` ?
4. finissant par `er` en fin de ligne ?

Raccourcis

Accolades

Les accolades indiquent un intervalle de répétitions :

- `lo{1,3}ng` reconnaît long, loong et looong
- `lo{3,}` reconnaît looong, loooong, etc.
- `lo{3}ng` ne reconnaît que looong

Exercice : écrire une expression régulière qui reconnaît exclusivement `lool`, `looool` et `loooool`.

Classes de caractères

Classe	Signification	Équivalent ASCII
`\d`	un chiffre	`[0-9]`
`\D`	tout sauf un chiffre	`[^0-9]`
`\w`	les caractères alphanumériques et _	`[a-zA-Z0-9âàéèêëôûüù_]`
`\W`	tout sauf les caractères alphanumériques et _	`[^a-zA-Z0-9âàéèêëôûüù_]`
`\s`	une espace	espace + `\t` + `\n`…
`\S`	tout sauf une espace	…

Notes :

- En général `\X` reconnaît le contraire de `\x`
- Le sens de `\w` et `\s` peut varier suivant les moteurs de regex

Exercices

Faire les quatre premiers niveaux sur : https://alf.nu/RegexGolf

1. Warmup
2. Anchors

3. It never ends
4. Ranges

Pour s'entraîner

- Tester ses regex en ligne : https://regex101.com
- Exercices : https://alf.nu/RegexGolf
- Mots croisés de regex : https://regexcrossword.com

Compléments

Stratégie de reconnaissance

- Par principe, un quantificateur est **glouton** (*greedy*) : il reconnaît le maximum de caractères possibles.
- En ajoutant `?` après un quantificateur, on le rend *lazy* : il reconnait le minimum de caractères.

Par exemple, si l'entrée est `AHHHHHHHHH` :

- `AH+` reconnaît la plus longue suite de caractères : `AHHHHHHHHH`
- `AH+?` reconnaît la plus petite suite de caractères : `AH`

Références

On peut utiliser des références numérotées pour rappeller le contenu d'un groupe :

`(spam|egg), sausage and \1` reconnaît :

- `spam, sausage and spam`
- `egg, sausage and egg`
- Mais pas `egg, sausage and spam`