Décrire et manipuler un document numérique

Cours 3 : Techniques de recherche avancées

L. Grobol <lgrobol@parisnanterre.fr>

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

- Formellement : définissent un test d'acceptabilité pour une suite (chaîne) de caractères : « est-ce que cette chaîne de caractère suit ce motif/ce patron/cette expression ? »
- Outil de recherche dans des (fichier) textes : « donne-moi toutes les sous-chaînes de caractères de ce fichier qui sont acceptables pour cette expression ».
- Utilisées pour extraire rapidement des informations à la syntaxe régulières dans des grandes masses de données :
  - Dates
  - Adresses
  - Structures linguistiques

Outils :

- IDE: VSCode, éventuellement dans sa version en ligne.
- Éditeur de texte brut : Notepad++ (Windows), Kate/Gedit (Linux), CotEditor (Mac)…
- Test en ligne : https://regex101.com/, https://pythex.org/

Possible, mais moins pratique pour ce cours :

- Traitement de texte : Google Drive, Word, Writer, Pages…
- 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. Accéder à la fonction de recherche (en général ctrl+F/R/H).
    - Cochez « expression régulière » (ou `.*`) et « respecter la casse » (ou `Aa`)

Principe général

- Règle générale : par défaut chaque caractère d'une expression correspond (*match*) à ce caractère :
    - `France 2` match `France 2`, mais pas `France2` ni `France 3`
    - (en mode `Aa`) `madeleine` match `madeleine`, mais pas `Madeleine`
- 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 »

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

Exercice : que matchent les expressions suivantes ?

1. `a?typique`
2. `pomm?es`
3. `lo*l`
4. `loo*l`
5. `lol+`

(Tester vos réponses dans un éditeur)

Caractères spéci · aux

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)
`\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. `cap$`
3. `br.n`
4. `^cap`
5. `premier paragraphe\ndeuxième paragraphe`

Déspécialisation

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

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

- `3\.0` matche `3.0`, mais pas `310`
- `price: \$5` : matche `price: $5`
- Pareil pour `^`, `[`, `]`, `(`, `)`, `*`, `+`, `?`, `{`, `}`

Et pour représenter un anti-slash ? Pareil

- `machin\\truc` matche `machin\truc`

Les opérateurs

Alternatives

La barre droite `|` indique une alternative :

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

On les utilise le plus souvent avec des parenthèses pour contrôler sa portée

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

Ensembles de caractères

Les crochets indiquent un ensemble de caractères à matcher :

- `[abc]` matche un caractère qui est soit `a`, soit `b`, soit `c`
- Avec un tiret : `[a-r]` matche les lettres de `a` à `r`
- `[a-z]` matche les lettres de l'alphabet en minuscules
- `[a-zA-Z]` matche aussi les majuscules*

Exercice :

Que matchent :

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

Écrire une expression régulière qui matche 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

- `[^abc]` matche tout sauf `a`, `b` et `c`
- `[^0-9]` matche tout sauf des chiffres
- `[^A-Z]` matche tout sauf des majuscules sans diacritiques

On peut les combiner avec les quantificateurs :

- `[aeiouy]+` matche une suite de voyelles : `a`, `aa`, `ae`, `iey`…
- `[^abc]*` matche une suite de caractères différents de `a`, `b` et `c`.

Groupes

Les parenthèses permettent de construire des groupes

- `(ab)+` matche une suite de `ab` : `ab`, `abab`, `abababab`…
- `([ab]c)+` matche `ac`, `bc`, `acac`, `acbc`, `bcac`…
- `([aeiouy][^aeiouy])+` matche une suite de couples voyelle—(non-voyelle).

Exercice : que matchent les expressions suivantes ?

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

Exercice

Dans le fichier `clg.txt`, combien trouvez-vous de matchs pour chacun des expressions régulières suivantes ? Que signifient-elles ?

1. `.évolution`.
2. `[a-zA-Z]évolution`
3. `[^rR]évolution`

Raccourcis

Accolades

Les accolades indiquent un intervalle de répétitions :

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

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

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	…
`\b`	frontière de mot (début ou fin de mot)	`\bjour` : `journée` mais pas `ajouré`

Notes :

- En général `\X` reconnaît le contraire de `\x`
- Le sens de `\w`, `\s` et `\b` peut varier suivant les moteurs de regex, suivant leur support d'Unicode. En particulier ils marchent mal dans VSCode.

Propriétés Unicode

Classe	Signification
`\p{L}`	une lettre
`\P{L}`	tout sauf une lettre
`\p{P}`	Une ponctuation
`\P{P}`	tout sauf une ponctuation
`\p{Lu}`	une lettre majuscule
`\p{Ll}`	une lettre minuscule

Voir les détails sur MDN et la liste des propriétés du type « catégories générales » (classes de caractères) du consortium Unicode.

Exercices

Dans ‘clg.txt, 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 ?

Faire les niveaux suivants sur https://alf.nu/RegexGolf

- *Warmup*
- *Anchors*
- *Ranges*

Faire le tutoriel et les 5 premiers puzzles de https://regexcrossword.com

Pour s’entraîner

- Tuto interactif : https://regexlearn.com/learn/regex101
- Cours et exos : http://regextutorials.com

Compléments

RegEx et automates

Au tableau ! Voir aussi https://cyberzhg.github.io/toolbox/nfa2dfa

Stratégie de reconnaissance

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

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

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

Références

On peut utiliser des références numérotées pour rappeler 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`