

TP: Crawler

Jean-Claude Moissinac

jean-claude.moissinac@telecom-paristech.fr

inspiré d'un TP de Pierre Sennelart

17 mai 2018

Dans ce TP, nous allons nous intéresser au site FPO <http://www.freepatentsonline.com/>. Le but va être d'extraire des informations sur des brevets sur ce site de description de brevets.

Nous allons exploiter différents outils pour réaliser notre crawl ; du plus simple au plus complexe, wget, Tidy, HTML5Lib...

A la fin du TP, ou, au plus tard vendredi 25/5, vous m'enverrez par email une archive portant votre nom. Elle comportera les sources de vos programmes nommés comme spécifié dans le sujet et un fichier README.txt où vous expliquerez l'organisation des fichiers fournis, les choix d'informations que vous avez cherché à obtenir et des indications sur les informations réellement obtenues.

ATTENTION : lors des appels répétés à un même site et compte tenu du nombre que vous êtes, respectez un délai d'au moins 5 secondes entre 2 appels ; le non-respect de cette règle risque d'entraîner le blocage de nos accès

(des outils comme lxml, Scrapy, Selenium, BeautifulSoup pourront être utilisés optionnellement)

Installations préliminaires

Le TP peut être réalisé sur tout ordinateur comportant une installation de Python 3.x, Firefox, wget, ainsi que les paquets Python `tidylib` et `html5lib` (installables via les commandes `pip install` ou `easy_install`). Il est vivement recommandé de créer un environnement virtuel avant de faire l'installation des outils:

```
python3 -m venv ./nomDossierPourVotreEnvironnement
source nomDossierPourVotreEnvironnement/bib/activate
```

1 Crawls élémentaires avec wget

wget est un outil en ligne de commande permettant d'effectuer des crawls élémentaires. Dans sa forme la plus simple « `wget url` », il permet de récupérer une page Web à l'URL `url`. Vous pouvez consulter la documentation de wget avec `<man wget` ».

1. Téléchargez la page <http://www.freepatentsonline.com/8591332.html> (description d'un brevet) avec wget. Ouvrez le fichier ainsi téléchargé et comparez-le à la page Web d'origine. Qu'observez-vous ?
3. Avec les options « -r --wait=1 », wget permet de télécharger de manière récursive un ensemble de pages Web, avec un délai de *politesse* d'une seconde entre deux requêtes vers le même serveur. Essayez cette option pour télécharger un ensemble de descriptions de brevets depuis FPO. Que constatez-vous ? Vous pouvez interrompre le crawl avec CTRL+C.
4. wget dispose d'une option « -np » permettant de ne télécharger que les pages Web contenues dans le même répertoire que la page Web initiale (ou dans un sous-répertoire de celui-ci). Essayez de mettre en œuvre cette option pour télécharger des articles de FPO, notamment celui de la partie 1. Vous devriez parvenir à récupérer une partie d'entre eux, mais il reste des requêtes *inutiles* faites par le crawler. Commentez.
5. wget dispose d'une option « --reject-regex » permettant de ne pas télécharger les pages correspondant à un certain motif (expression régulière); utiliser cette option pour exclure des pages qui vous semblent inutiles en indiquant quel(s) motifs de caractère(s) les URL que vous souhaitez rejeter comporte(nt) (attention, si vous utilisez des caractères spéciaux pour les expressions régulières, il faut les préfixer d'un antislash). Laissez le crawl travailler un moment : combien avez-vous récupérés de description de brevets ? Réussissez-vous à ne récupérer que des descriptions de brevet ? Si oui, avec quelle expression régulière ?

2 Crawl systématique par programme Python

Nous allons maintenant utiliser Python3 pour faire un crawl plus systématique des articles qui nous intéressent. Téléchargez le fichier tasktimer.py et un exemple d'utilisation dans etape0.py à ces adresses :

<https://perso.telecom-paristech.fr/moissina/TPCRAWL/etape0.py>

<https://perso.telecom-paristech.fr/moissina/TPCRAWL/modules/tasktimer.py>

(ici vous pouvez aussi envisager d'utiliser scrapy)

1. Commencez par faire un programme minimal qui fait la requête
http://www.freepatentsonline.com/result.html?sort=relevance&srch=top&query_txt=video&submit=&patents=on
sur le site FPO pour charger (par exemple en utilisant urllib) et sauver dans un fichier la réponse du serveur FPO à cette requête. Cette requête interroge le serveur pour obtenir une liste de brevets concernant la vidéo. Appelez le programme etape1.py
2. Faire un programme –inspiré des programmes etape0 et etape1- qui charge une liste de pages. Appelez ce programme etape2
Maintenant, au lieu de sauver le contenu reçu, vous allez commencer à analyser le contenu. Pour commencer vous pouvez utiliser pytidylib pour corriger les défauts éventuels des pages chargées (optionnellement, si votre compte permet d'utiliser

BeautifulSoup, vous pouvez faire les choses avec cet outil). Ensuite, vous allez utiliser `html5lib` pour accéder à des parties de la page. Avec le XPATH `'.///xh :a'`, récupérez tous les tags `<a>` présents dans la page (xh est le préfixe associé au namespace

<http://www.w3.org/1999/xhtml>)

Combien y a-t-il de liens dans la page initiale puis dans d'autres pages que votre programme visite?

Ajoutez chaque lien trouvé à la liste des liens à charger.

Ajoutez une condition d'arrêt lorsque 30 pages html ont été chargées (vous pouvez tester avec d'autres valeurs que 30).

A ce moment-là, combien de pages ont été visitées ? Combien de liens sont dans la file d'attente des liens à traiter ?

Donnez le nom `etape3.py` à ce programme.

3. Ajouter au programme précédent du code pour éviter de charger plusieurs fois la même URL. Programme `etape4.py`
4. Ajouter au programme précédent du code
 - pour déterminer si une page est une page de description d'un brevet (+> une page qui nous intéresse)
 - pour estimer empiriquement si un lien mérite d'être suiviProgramme `etape5.py`
5. A l'aide de `html5lib` et d'expressions régulières sélectionnez certaines URL qui respectent certains modèles (vous pourrez mettre au point vos expressions régulières à l'aide de <https://regex101.com/>). Programme `etape6.py`. Vous décrirez quels liens sont supposés sélectionner vos expressions.
6. Choisissez d'autres informations dans ces pages dont vous allez chercher à récupérer la valeur (nom d'auteur de brevet ? entreprise ?...). Sauvez les données obtenues dans une structure json. Programme `etape7.py`
7. Mettez en place un processus qui va chercher, sur les principes précédents, dans FPO, au moins 3 brevets dont un des inventeurs est français et qui a à voir avec le multimédia. Programme `etape8.py`. Vous indiquerez comment lancer le processus et créerez un fichier nommé `multimedia.txt` qui contiendra les 3 URLs des pages décrivant ces brevets, chacune sur une ligne.

Pour aller plus loin sur ce sujet, pour développer un crawler en Python, nous vous recommandons `scrapy`.