

Amélioration : respecter robots.txt

Mot d'introduction

Cette feuille propose un objectif supplémentaire pour le projet (secondaire par rapport aux autres). Il s'agit de faire en sorte que notre moissonnage de pages web respecte les fichiers `robots.txt` disponibles sur certains serveurs. Nos programmes seront ainsi plus conformes aux bonnes pratiques.

Si un fichier `robots.txt` est disponible à la racine d'un site web, (par exemple à l'adresse <https://monsuper.site.fr/robots.txt>), les directives qu'il indique concerneront les pages de tout ce site (toutes les pages dont l'url commence par `monsuper.site.fr` dans l'exemple).

Vous pouvez déjà observer un tel fichier dans le dossiers `/docs/` de notes git. Il s'agit de celui de Wikipedia.

Pour simplifier le travail, on ne s'occupera que de deux directives : “**User-Agent :**” et “**Disallow :**”. La première indique quel robot est ciblé par les règles qui suivent et la seconde indique un chemin qui ne doit pas être moissonné, ni le chemin exact ni aucun des sous dossiers et fichiers contenus dans le sous-arbre correspondant.

Certains robots bien connus on en effet un **User-Agent** clairement identifiable (information envoyée dans les en-têtes des requêtes HTTP), mais on peut aussi désigner l'ensemble des clients web par la directive **User-Agent : ***.

Nous devons donc nous sentir concernés par toutes les directives **Disallow : /...** qui suivent la directive **User-Agent : *** et ignorer les autres (qui généralement la précèdent).

(Pour en savoir plus, voir aussi <https://en.wikipedia.org/wiki/Robots.txt>)

La suite de cette feuille contient des indications pour intégrer la prise en compte de ces fichiers à votre projet.

Exercice 1 Stratégie générale

Tout le projet part du traitement d'une liste d'urls.

La stratégie que l'on vous propose de suivre est la suivante : écrire un nouveau script bash qui va construire une liste d'URL interdites et sauvegarder cette liste dans un fichier (une *blacklist*) qui ira de paire avec le fichier de liste d'url initiale.

Une fois la *blacklist* créée, notre programme principal pourra l'utiliser pour vérifier avant chaque requête `curl` que le moissonnage de l'url n'est pas interdit.

Exercice 2 Construction d'une *blacklist*

On peut déjà remarquer que dans nos listes d'urls, plusieurs urls peuvent se trouver sur le même serveur et ainsi partager le même fichier `robots.txt`. Il faudra ainsi tronquer les urls pour n'avoir que l'adresse du serveur et une étape de dédoublement sera donc attendue (la commande **uniq** pourra être utile).

Il faudra ensuite **pour chaque serveur** aller chercher (avec **curl**) le contenu du fichier :

`<url du serveur>/robots.txt`.

Pour faciliter le traitement, on peut sauvegarder le `robots.txt` dans le dossier temporaire `/tmp/`. On traitera ensuite **chaque ligne** d'un fichier `robots.txt` pour extraire les urls interdites (on reconstruira l'url complète `<url du serveur>/<chemin interdit>`).

Il faut bien penser à **ignorer** toutes les lignes tant qu'on n'a pas atteint la ligne contenant la directive **User-Agent : *** et n'agir qu'après avoir vu celle-ci. On peut pour cela utiliser une variable supplémentaire pour indiquer si **oui ou non** on a atteint la zone du bon User-Agent.

Si la variable du User-Agent est à **oui** et qu'on trouve une ligne contenant une directive **Disallow : ...**, il faut alors ajouter l'url interdite à notre *blacklist*.

L'ensemble de toutes ces adresses interdites de tous les fichiers `robots.txt` seront consignées dans une *blacklist* correspondant à notre fichier d'url initial. On pourra par exemple sauvegarder la blacklist correspondant à `URLS/fr.txt` dans un fichier nommé `URLS/fr.txt-blacklist` afin de le retrouver facilement.

Exercice 3 Prise en compte des *blacklist* et ajout au tableau

On doit maintenant modifier notre programme principal en ajoutant la vérification pour chaque URL que celle-ci ne se trouve pas dans la *blacklist*.

On pourra aussi modifier nos tableaux de résultats en ajoutant une colonne qui indique si un fichier `robots.txt` interdit ou autorise le moissonnage de chaque adresse. Idéalement, on distinguera les cas où le fichier `robots.txt` est présent et nous autorise l'accès, des cas où le fichier n'existe pas (et où l'accès est donc autorisé par défaut).