Sign-of-life crawler

This paper describes the approach used by the crawler. Last updated on 18/10/2019.

Agenda

[I -MAIN APPLICATION 2](#_Toc20424202)

[1) Collecting domain data 2](#_Toc20424203)

[2) Parking classification 2](#_Toc20424204)

[3) The parking pattern 5](#_Toc20424205)

[II - LANGUAGE EXTENSION FOR PARKING DETECTION 7](#_Toc20424206)

[III- LIMITS AND IMPROVEMENT DIRECTIONS 8](#_Toc20424207)

[1) False positive for parking patterns 8](#_Toc20424208)

[2) Languages 9](#_Toc20424209)

[3) External JavaScript redirection 9](#_Toc20424210)

[4) Texts in images 9](#_Toc20424211)

## I -MAIN APPLICATION

### 1) Collecting domain data

* Collect zonefiles of a given TLD and convert it into a csv file of all URLs to visit.
* Visiting each URL of the above TLD using the following format of full URL: http:// + URL (without www.). Based on samples analysis, 50,000 URLs will yield a confidence interval below 1 % range.
  + Python, like many other general purpose codes and like all browsers (Chrome, Firefox…), has HTTP client libraries. So far, “aiohttp” is used. This library allows communicating on the internet by sending messages following the HTTP or HTTPS protocols. In particular, the standard GET method will yield the content of the main page of an URL website, which is an HTML page. This method is also used by browsers when we type the URL. With this method, the code will send GET messages to all websites of the above TLD.
* In DEBUG mode, it stores the content of the website as well as some server interactions (e.g. redirections) for later use

URLs with correct response will return an HTML page as well as information about the server interactions (History of redirections and status codes). The next two steps are performed:

* Extracting the displayed text from the HTML and storing it. The ALT text of images as well as the page description tag (<meta name=”description”>) are added to this displayed text.
  + An HTML page contains HTML, CSS, JavaScript and PHP codes. It defines precisely the content and format of a website. In particular, some of the displayed text is directly written on the page, like “this domain is parked”. To collect this text, some scraping libraries help to partially interpret the HTML code. It will identify tags, for example the tag “h1” is often used for titles of a webpage and “p” for paragraphs. So far, the library “beautifulSoup” is used. At the date of writing, JavaScript is not interpreted, but might be implemented in a later development (parked domains are often very simple websites).
* Identifying the language of the website preceded by a filtering of acronyms, codes and numbers
  + A simple natural language processing (NLP) library, called “langdetect” developed and open sourced by Google (Apache License 2.0), allows identifying the language of the displayed text. The recent progress in NLP provides access to more advanced approaches for this task.

With all this collected information, the classification models can begin: So far, a parking classification is implemented.

### 2) Parking classification

The classification follows the decision tree below. The summary of classification criteria by category is provided in Table 1 .

#### Errors

* If the HTTP Client fail to identify the IP address of a URL, the URL is categorized as **Connection Error**.
* If the IP address is found but the corresponding server fails to give a valid answer, the URL is categorized as **Invalid Response**.
* If the server provides a response with a status code 301, 302, 307, 308, the domain is redirecting to another page. This page is visited and the classification parking is reinitialized. A redirection flag keeps track of this operation.
* If the server provides a response with status code 401-2-3, we are not authorized to communicate to this server. If the code 401-2-3 persists, the URL is categorized as **Unauthorized**.
* If the server answers with other 4XX (client errors) or 5XX (server errors) code, the URL is categorized as **HTTP Error**.

#### Valid responses

The server provides a 2XX code with an HTML page

1. The HTML page is parsed to extract specific information related to links to other websites, as well as type of content in the page:
   * “a” and some “area”: this tag indicates a link to another website/page/object
   * “base” tag: reference URL for all relative links
   * “frameset”, ”frame” and ”iframe” : HTML tag to display a page from a different location
   * “window.location” : similar to iframe but with JavaScript code
   * The META tag with the key-value “http-equiv=refresh”: another way of redirecting
   * Presence of JavaScript code with the “script” and “noscript” tags. In particular, this code can include:
     + Redirections through the function “window.location”
     + Dynamically written HTML code with “document.write”
     + A note on the necessity to use javascript to collect the full content of the page
   * Quantity of non-table, non-list and non-formatting HTML tags
   * Quantity of displayed words and letters
   * Quantity of lines in the normalized html (without comments and multi-line breaks)

The second part of the tree can now be implemented:

* if the domain of the URL of the response is different than the initial domain 🡪 the target page is used for classification.The redirection flag is raised (*redirection\_type* = http)**.**
* Following the redirection tree in the picture below, if there is only one significant meta-refresh AND/OR frameset and frame OR iframe OR window.location(.href), the HTML page contains less than 150 lines and the link leads to another domain. Here “significant” means the link is not empty/blank.html/notfound and the element is not included in basic content tags like P, TR, TL, SPAN… 🡪 the redirection target page is visited and used for classification.The redirection flag is raised.
* If a link to one and only one registrar website is found, the website might be a registrar. However some registrars also provide website building services (ex: Wordpress) and other registrars have a diversified business portfolio (like Microsoft). To filter out the first category, a restriction on the quantity of tags is applied. Generated websites tend to have a complex HTML structure. For the second category, a website that displays a lot of other contents must also have a parking note (cf next bulletpoint) to be considered in this category. A domain that satisfies these conditions is classified as **Parking Response Registrar.** The registrar that are considered are listed in the Table-2.
* The displayed text is analyzed to find a parking note (=”parking pattern”) like “this domain is for sale”. If such note is found, the domain is classed as **Parking Individual Content**. The details on how notes are identified is provided in part I - 3.
* If the page includes a “Javascript required” note or “document.write”🡪 the page is revisited with a JavaScript interpreter. So far, we use Chrome and its driver through the Python selenium library.
* If the displayed text has less than 5 words and has no JavaScript, no frames and no links, the URL is considered as **Blank Page.**
* If the page does not fit any of the above criteria, it is considered as **Normal**.

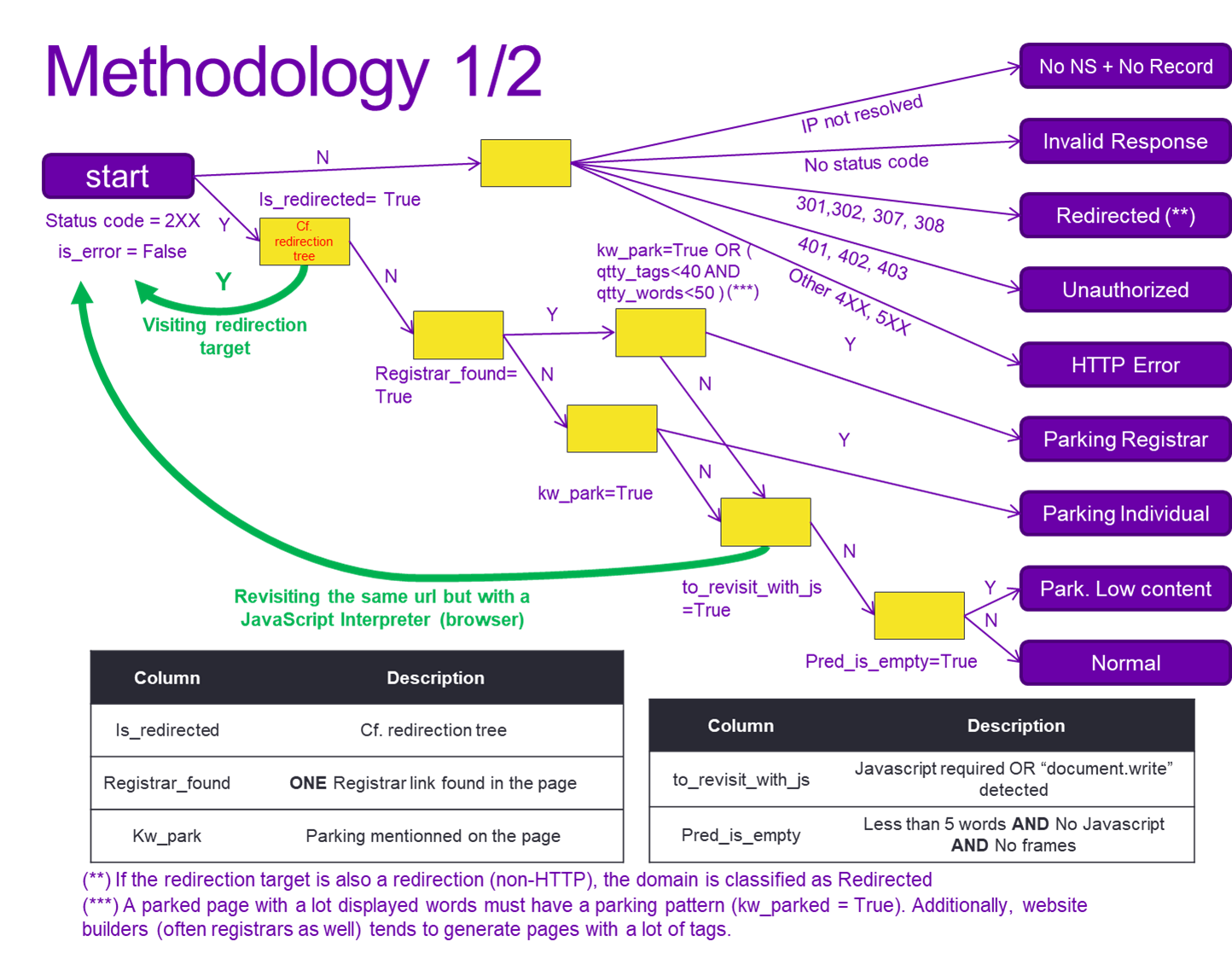


Figure 1 - Parking decision tree

Table - Parking Category classification criteria

|  |  |  |
| --- | --- | --- |
|  | Category | Classification Criteria |
| Error | Connection Error | IP address is not resolved |
| Invalid Response | IP address is found but the corresponding server fails to give a valid answer |
| Unauthorized | Server responds with a 401, 402 or 403 code |
| HTTP Error | Server responds with a 4XX (client errors) or 5XX (server error) code |
| Parking | Parking Response Registrar | Websites with only one registrar link, a simple structure and a limited number of hard-coded displayed words. For bigger website to fall in that category, a parking note must be identified (cf. Parking Individual Content) |
| Parking Individual Content | The displayed text is analyzed to find a parking note like “this domain is for sale” |
| Blank Page | Displayed text is less than 5 words long, No Javascript, No frames and No links |

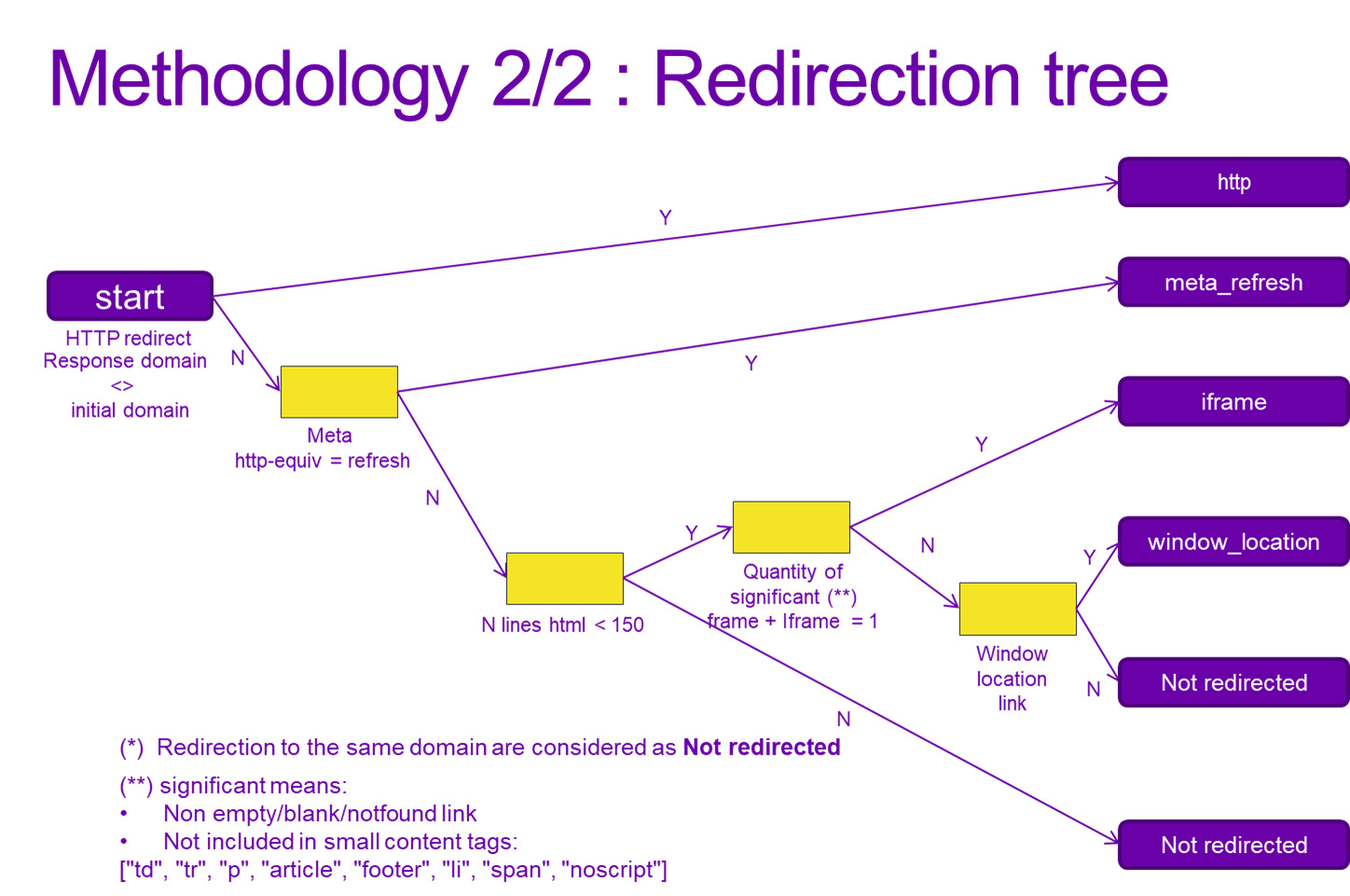


Figure 2 - Redirection tree

Table - List of detected registrars/hosting companies

|  |
| --- |
| Registrar |
| ICANN members |
| CENTR members |
| Russia |
| New Zealand |
| Estonia |
| Montenegro |

### 3) The parking pattern

#### 1) Definition

In all visited websites, a parking notice (like “this website is for sale” or “This is a parked domain”) always follow a simple pattern described in the picture below. The pattern is a core key word and an attribute key word in the same sentence.

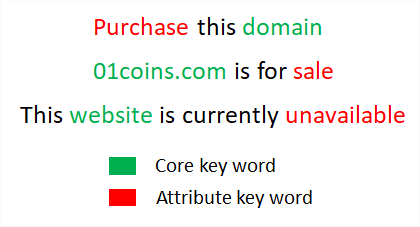


Figure 3 - parking pattern

To detect this pattern, Regular Expressions (=RegEx) are used. This library allows to find a sequence of characters in a text. For example, the regex “sales\*” will allow to detect both the keywords “sale” and “sales”. A more advanced use is implemented for sentences disambiguation like “www.abc.com is reserved” and “www.abc.com All rights reserved”. The first sentence is clearly a parked notice but not the second one. To handle this case, the regex “ (?<!rights) reserved” will detect all the words “ reserved” that are not preceded by “rights”.

Finally, some attribute key words are specific to one or more core key words. For example, the attribute “register” indicates a parked domain when it is associated with the core “domain”, but not with “website” (= login).

These patterns are different for each language. For now, the languages in the table below are supported. To derive the patterns of a new language, the process is described in part II of this document.

Table 3 - Supported languages

|  |  |
| --- | --- |
| language | code |
| English | en |
| Russian | ru |
| Chinese | zh-cn |
| Chinese | zh-tw |
| Spanish | es |
| French | fr |
| Swedish | sv |
| German | de |
| Estonian | et |
| Danish | da |
| Italian | It |
| Slovak | sk |

#### Detection approach

The displayed text is analyzed in four steps:

* + Load the patterns (cores + attributes) for the identified language. In addition, English patterns are always searched by default, no matter the detected language
  + Find a core key word of the pattern
  + Extract the surrounding sentence. The definition of a sentence depends on the language.
  + Look up for any attribute key word in the sentence, before or after the core key word.

🡪If these four steps yield at least one full pattern, the site is categorized as **Parking Individual Content**.

## II - LANGUAGE EXTENSION FOR PARKING DETECTION

To add a new language, a Semi-Automatized extension has been developed. It uses a bottom up approach to identify the various sentences that indicates a parked page. The process to add a language includes the following steps:

* Translate the usual core keywords equivalent to “domain”, “website” and “page”. For Danish as example, one can use "*domæne*" (also captures “domænet”), "websted" and "*side*".
* In DICO\_CHARACTERS\_BY\_LGG (cf. figure 4), add the regular expression pattern that matches all characters for that language. This is available online. Indeed, each language has a set of specific characters (e.g. ñ in Spanish). This character set defines the sentence. Sentence separators must be excluded (points, question mark…). For Danish, it is “*[a-z '’\"æøå-]*”, wich includes A to Z, white space, quotation marks, dash and finally the three specific Danish letters (æ, ø and å).
* Extract the surrounding sentence for a given subset of URLs. Ideally the URLs should contain a lot of websites with the target language (AUTO). A sample of sentences is shown in figure 5.
* Count surrounding sentences across all these URLs and sort them by recurrence (AUTO)
* Translate the sentences using Google Translate (with Google Sheets function) for the English translation. At this step we get the table shown in figure 6.
* We go through all the translations with a significant cnt\_pattern (> 1% of domains visited). If it suggests a parking note, we select the most specific attribute key word among the words in the sentence. In case of doubt, the website is visited through a browser. Usually, translations of “parked” or “for sale” are the most popular attributes. For example, “Domænet er parkeret hos Bricksite” is a parking note and the word “parkeret” is added as attribute to the reference patterns of that language.
* Update the referential patterns in language\_patterns.py with the key words found.
* For maintainability, add the unique sentences to the test cases in test\_parking\_pattern.py. This will make sure that, whatever the changes of patterns in the future, the performances don’t regress.

For the current detailed list of patterns by language, please refer to the script language\_patterns.py. The exact sentences from which they are extracted, are listed in test\_parking\_pattern.py.

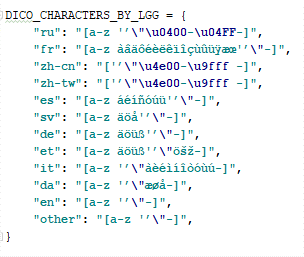


Figure 4 - characters used to delimit sentences for Russian, French etc…

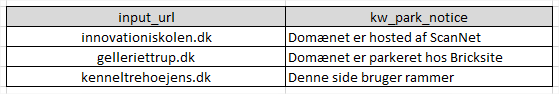


Figure 5 - sample of detected sentences for Danish

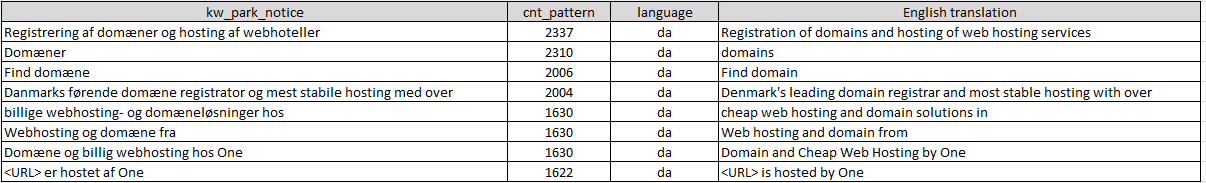


Figure 6 - Sentences sorted by frequency and translated

## III- LIMITS AND IMPROVEMENT DIRECTIONS

### False positive for parking patterns

Getting a comprehensive list of parking notice patterns is most likely impossible. Since millions of websites are to be visited, there will always be a few outliers with a specific phrasing. As long as the dominant practices are captured, we can detect 95% + of the parked domain with a notice.

To address this issue, the patterns have been built by giving priority to the most common sentences around the core keywords (domain, page, website, url.com). New patterns can be added to the referential patterns (language\_patterns.py) as soon as they are discovered. The test use cases will help prevent regression.

### Languages

As mentioned, for now, supported languages are listed in Table 3 of part I-3-1. To check relevance of results, a table sums up how many unsupported languages have been found at the end of a run. If too many websites are in an unsupported language, the crawler accuracy will deteriorate significantly.

### External JavaScript redirection

When a redirection is performed inside a JavaScript only code, the only way to detect it is to interpret all the webpage and its outbound links (all .JS files). To do that, a JavaScript interpreter, like a browser, is used. Currently if there is no indicator that JavaScript is required and if there is no “document.write” function, JavaScript will not be interpreted. For parking classification, we can assume that most of the redirection occurs in the main HTML page.

### Texts in images

Texts that are saved as pixels in an image are currently not considered. For this case, we rely on the text surrounding the image to classify the page. An OCR extension could be developed to extract the text of websites with only one image.