WP4-Corpus Acquisition and Annotation

ILSP

PANACEA 1st Technical Meeting
Athens, 15-16/04/2010

Outline

- Analysis of work
  - WP4.1-.2
  - WP4.3
- Issues for the WP3-6 Interoperability Session

WP4.1-.2

- Investigation for free/open-source tools that browse the Web by choosing the most promising links in order to maximize the relevancy of the retrieved pages to a specific topic

- Design and study of two candidate workflows appropriate for PANACEA needs.

WP4.1-.2

- Monolingual Focused Web crawler should find web pages relevant to a specific topic (in the selected language).
  - Input
  - Workflow
  - Output i) stored HTML files (UTF-8) , ii) XML file
  - Tools (a modified version of Combine seems promising)
- Bilingual Focused Web crawler should find pairs of web pages, relevant to a specific topic, from bilingual websites that each page in one language is translated to the other one.
  - Input i) seed terms in both languages, ii) seed URLs of bilingual sites
  - Workflow
  - Output i) stored HTML files (UTF-8) , ii) XML file iii) TMX file
  - Tools (a combination of Combine and Bitextor seems promising)
- Items to be discussed*

Input

Seed Term List

Example from Combine crawler

#This is the topic definition file

# Topic title=search engine

100: search project=OK

100: search result=OK

50: search engine=OK

Seed URL List

To construct such a list we could take advantage of the BootCat toolkit (suite of Perl scripts) [Baroni et al. 2004] .

1. Get terms
2. Create random tuples
3. Request Yahoo search engine for each tuple
4. Keep first 10 responses of each query

TO DO

Define the suitable XML schema for metadata (title and description of the topic, comments, the author’s name, date, etc (Is there a standard one?)

Back

Tools (Monolingual) 1/2

	Features/functionality	Input	Output	license
WebBootCat **BootCat front-end**	Applications of BootCat Toolkit (not crawlers). Send tuples of terms to yahoo search engine, get first 10 answers, download, cleaning based on Ntokens-Ntags, Greek not supported.	Term list URL list	Corpus	GPL
HTTrack	Designed for mirroring web sites. multithreaded, breadth-first, simple URLs filtering, language agnostic	URL list	Original HTML files	GPL
WIRE (C/C++)	multithreaded, PageRank, advanced URL filtering, HTML cleaning, encoding conversion, language detection	URL list	Normalized HTML files (UTF-8)	GPL
Heritrix (JAVA)	multithreaded, breadth-first, advanced URL filtering, language agnostic	URL list	Original HTML files	LGPL v.2.1
Combine (Perl)	multithreaded, combination of breadth-first and binary classifier (relevance to a topic), advanced URL filtering, HTML cleaning, encoding conversion, language detection (33, but not Greek)	Term list URL list	Normalized HTML files (UTF-8)	GPL

Tools (Monolingual) 2/2

	performance	Processing Speed	Feedback progress	Error handling	Documen-tation
WebBootCat **BootCat front-end**	4 terms →41 pages→ (143,883 words) in topic: Machine Translation	in 2.5min	Progress bar	fully integrated	in progress
HTTrack	-	N/A	Progress bar	fully integrated	full
WIRE	-	N/A	Updates a log file	fully integrated	full
Heritrix	4 URLs/sec (only 20 seed URLs) 20 URLs/sec (several hundred seed URLs)		Updates multiple log files	fully integrated	full
Combine	35% of visited pages are relevant	Handles up to 200 URLs/min	Updates a log file	fully integrated	full

Back

Output

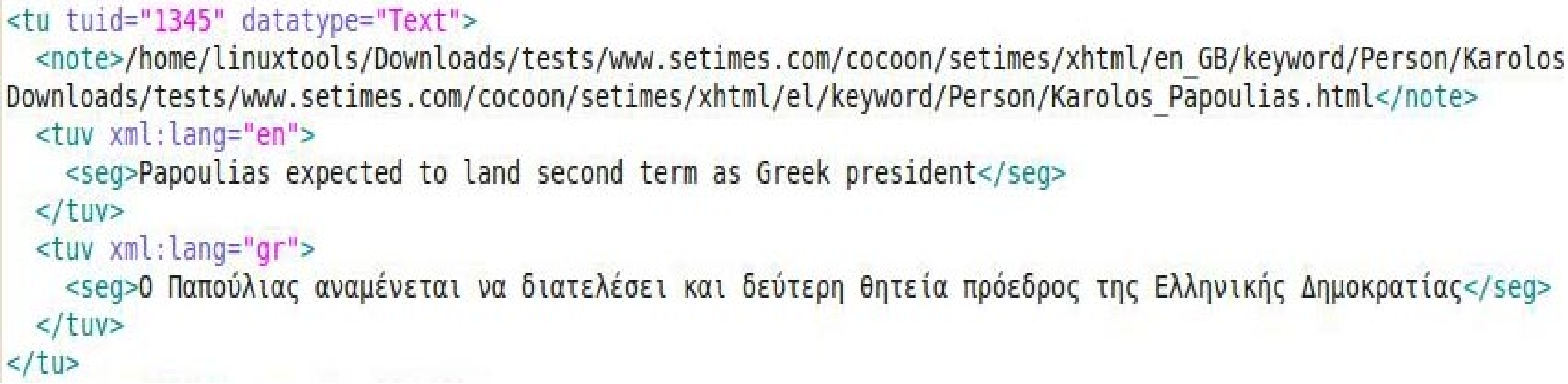
XML indicates the bitexts

Example from Bitextor's log file:

*22/3/110 14:17:33>> The bitext between /home/linuxtools/Downloads/tests/www.setimes.com/cocoon/setimes/xhtml/**en_GB**/ keyword/Person/Karolos_Papoulias.html and /home/linuxtools/Downloads/tests/www.setimes.com/cocoon/setimes/xhtml/**el**/keyword/Person/Karolos_Papoulias.html has been created>>*

Edit distance: 18.9207.

TMX indicates the aligned text blocks



```
<tu tuid="1345" datatype="Text">
  <note>/home/linuxtools/Downloads/tests/www.setimes.com/cocoon/setimes/xhtml/en_GB/keyword/Person/Karolos
Downloads/tests/www.setimes.com/cocoon/setimes/xhtml/el/keyword/Person/Karolos_Papoulias.html</note>
  <tuv xml:lang="en">
    <seg>Papoulias expected to land second term as Greek president</seg>
  </tuv>
  <tuv xml:lang="gr">
    <seg>Ο Παπούλιας αναμένεται να διατελέσει και δεύτερη θητεία πρόεδρος της Ελληνικής Δημοκρατίας</seg>
  </tuv>
</tu>
```

Back

Tools (Bilingual)

	Features/functionality	Input	Output	license
Bitextor (C/C++)	1) Filtering URLS and downloading pages, 2) feature extraction and comparison 3) classification of each pair as bitext or not. (All parameters for filtering and comparing can be configured via an XML file).	URL list	Html files (UTF-8) TMX with aligned text blocks	GPL

	performance	Processing Speed	Feedback progress	Error handling	Documentation
Bitextor (C/C++)	depends on the structure of each website. On a well structured website (Parliament of Canada), precision and recall were 99% and 85.33%. The corresponding values for an heterogonous one (University of Alacant) were 86% and 61%.	N/A	a message for each major step (e.g. downloading, comparing, generating bitexts) ; needs improvement	needs improvement	needs improvement

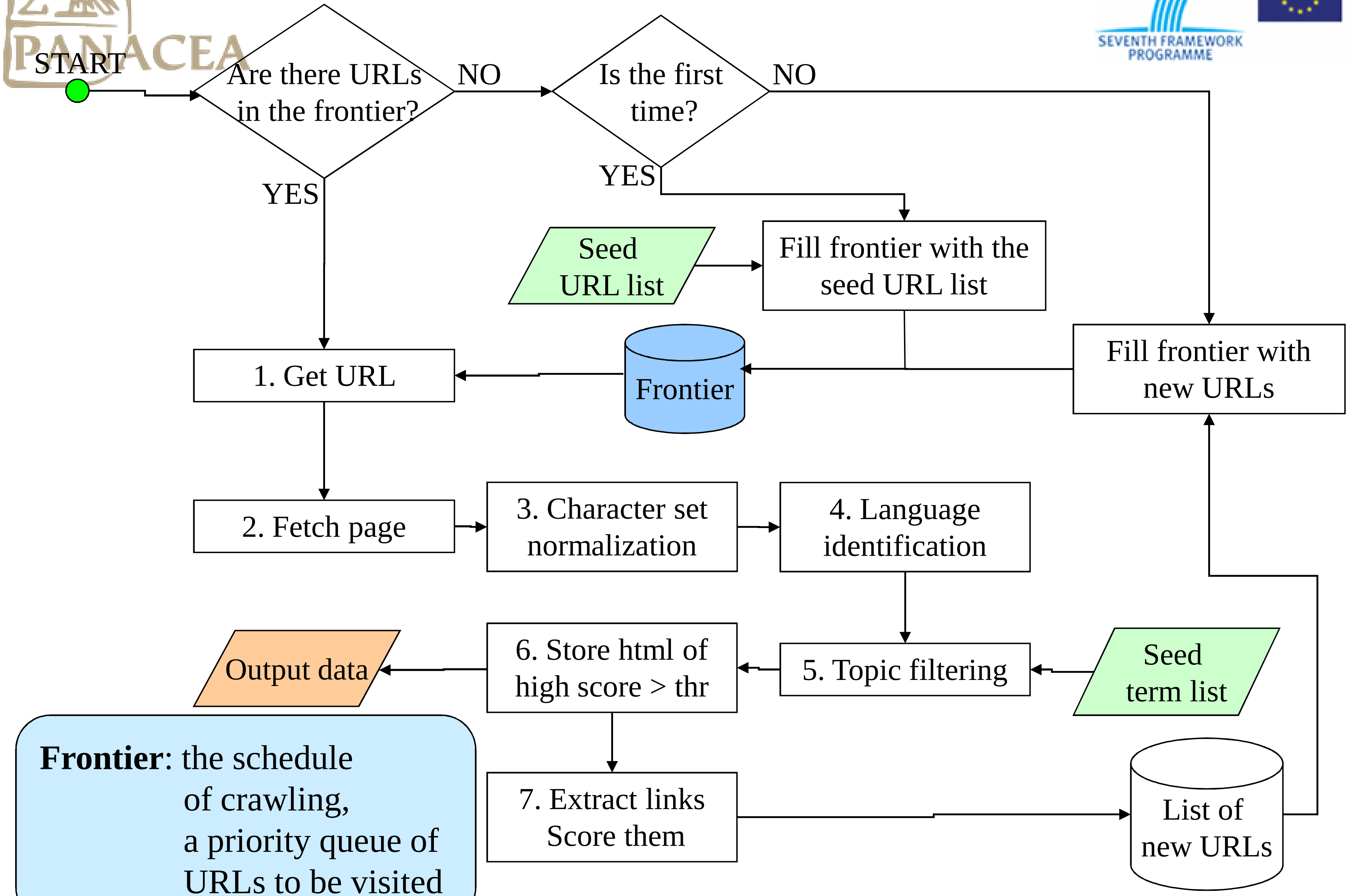
BACK

Discussion

- Topics for monolingual data acquisition
- Topics for bilingual data acquisition
- Language pairs for bilingual corpora
- Are there any URL lists of specific topics (as in Open Directory Project) for monolingual data crawling (i.e. to be used as the seed lists)?
- Are there any lists of URL pairs (as STRAND but in specific topics) for bilingual data crawling?

Back

Monolingual data focused crawling (Workflow)



START

Are there URLs in the frontier?

NO

Is the first time?

NO

YES

YES

Seed URL list

Fill frontier with the seed URL list

Fill frontier with new URLs

1. Get URL

Frontier

2. Fetch page

3. Character set normalization

4. Language identification

5. Topic filtering

Seed term list

6. Store html of high score > thr

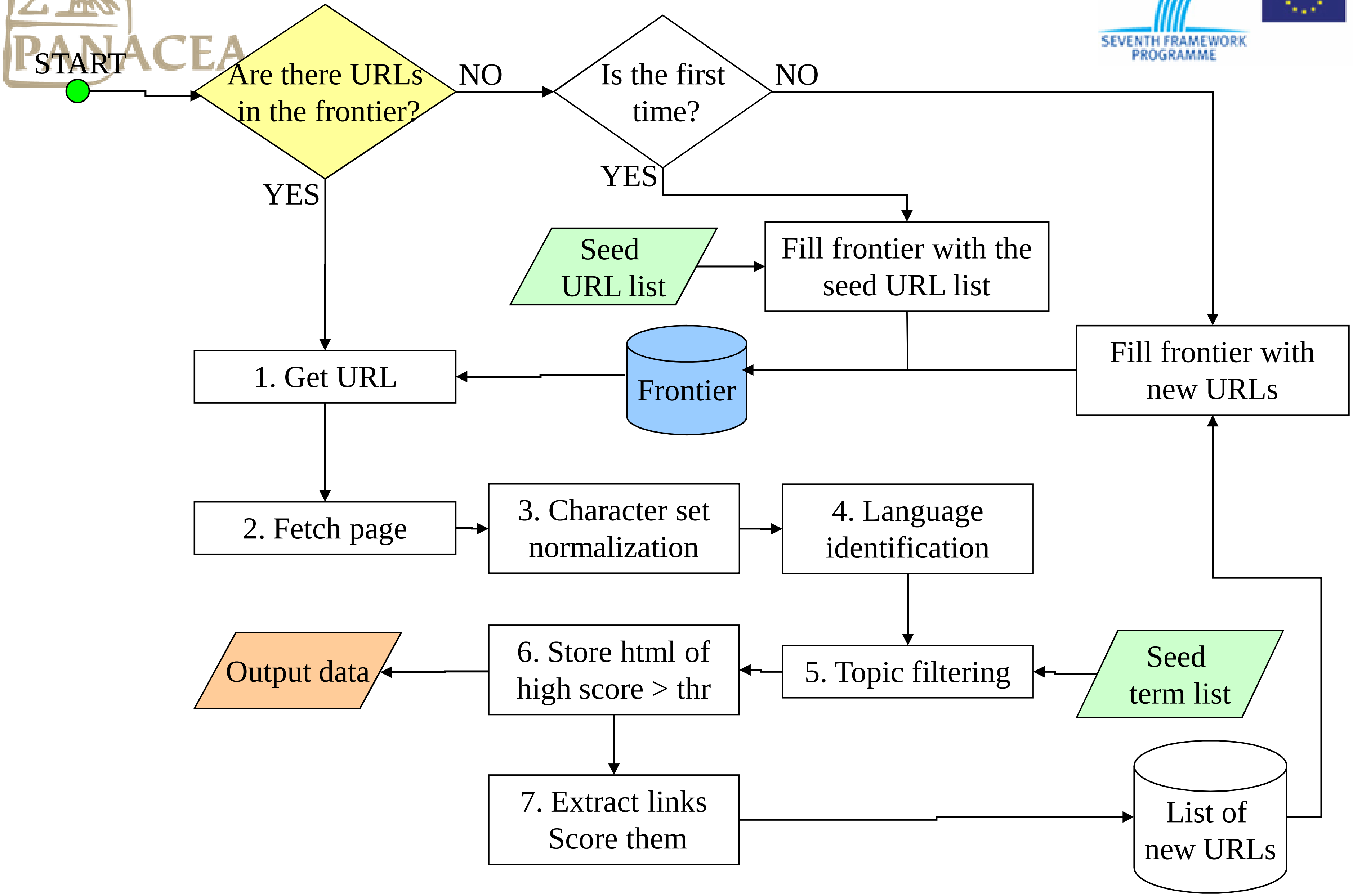
Output data

7. Extract links Score them

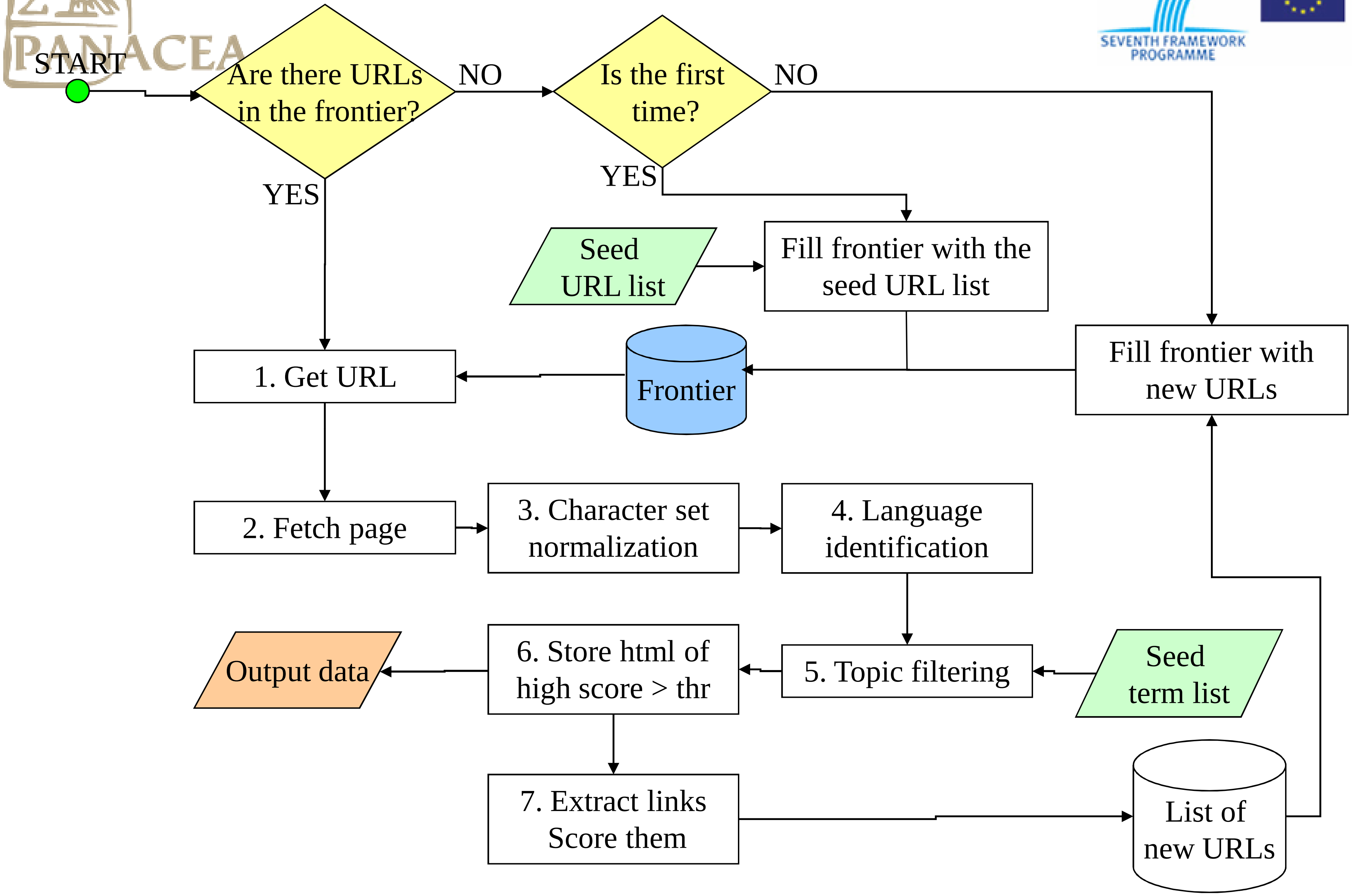
List of new URLs

Frontier: the schedule of crawling, a priority queue of URLs to be visited

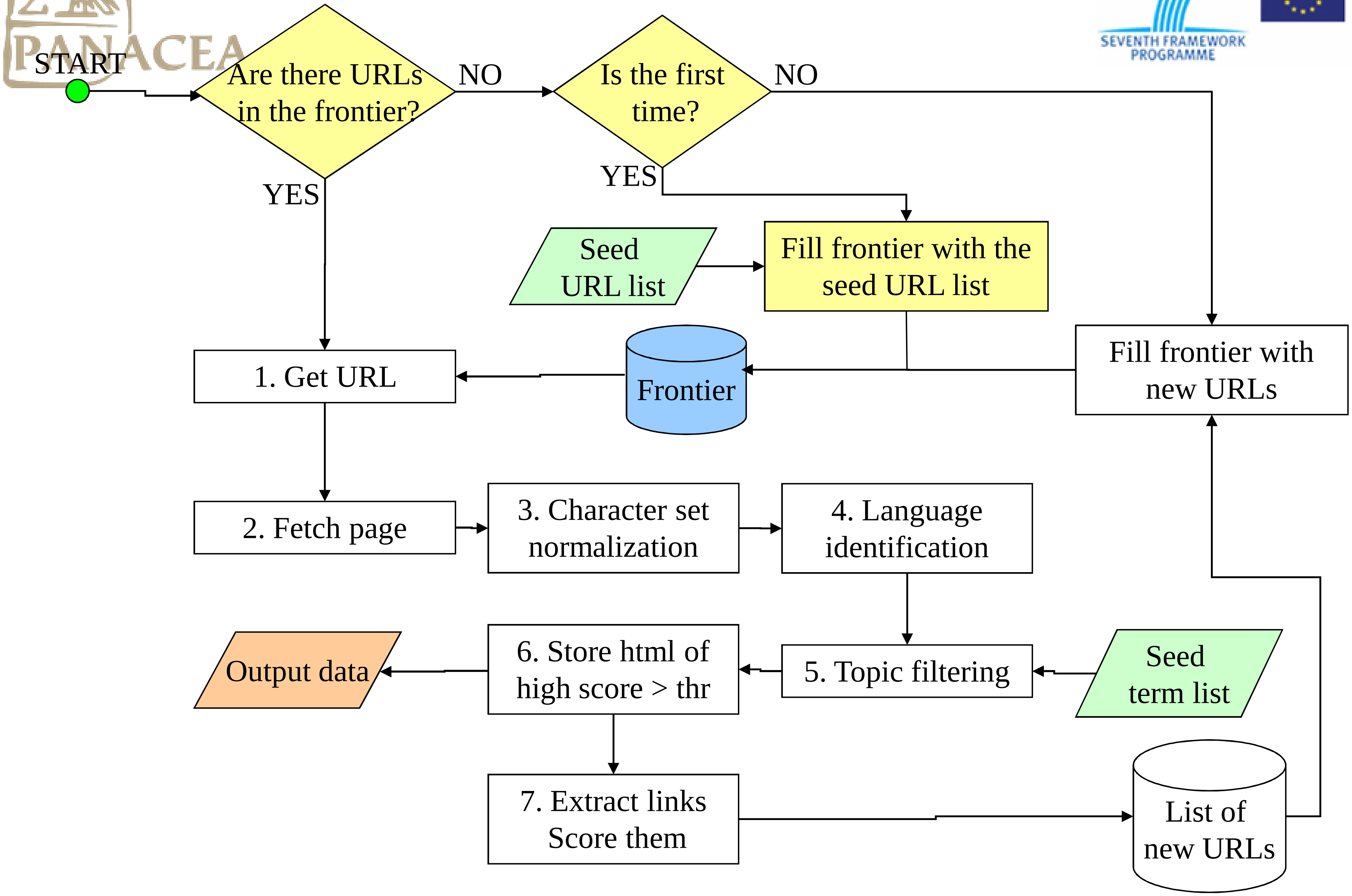
Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
YES
Is the first time?
NO
YES
Seed URL list
Fill frontier with the seed URL list
Frontier
Fill frontier with new URLs
1. Get URL
2. Fetch page
3. Character set normalization
4. Language identification
5. Topic filtering
Seed term list
6. Store html of high score > thr
Output data
7. Extract links Score them
List of new URLs

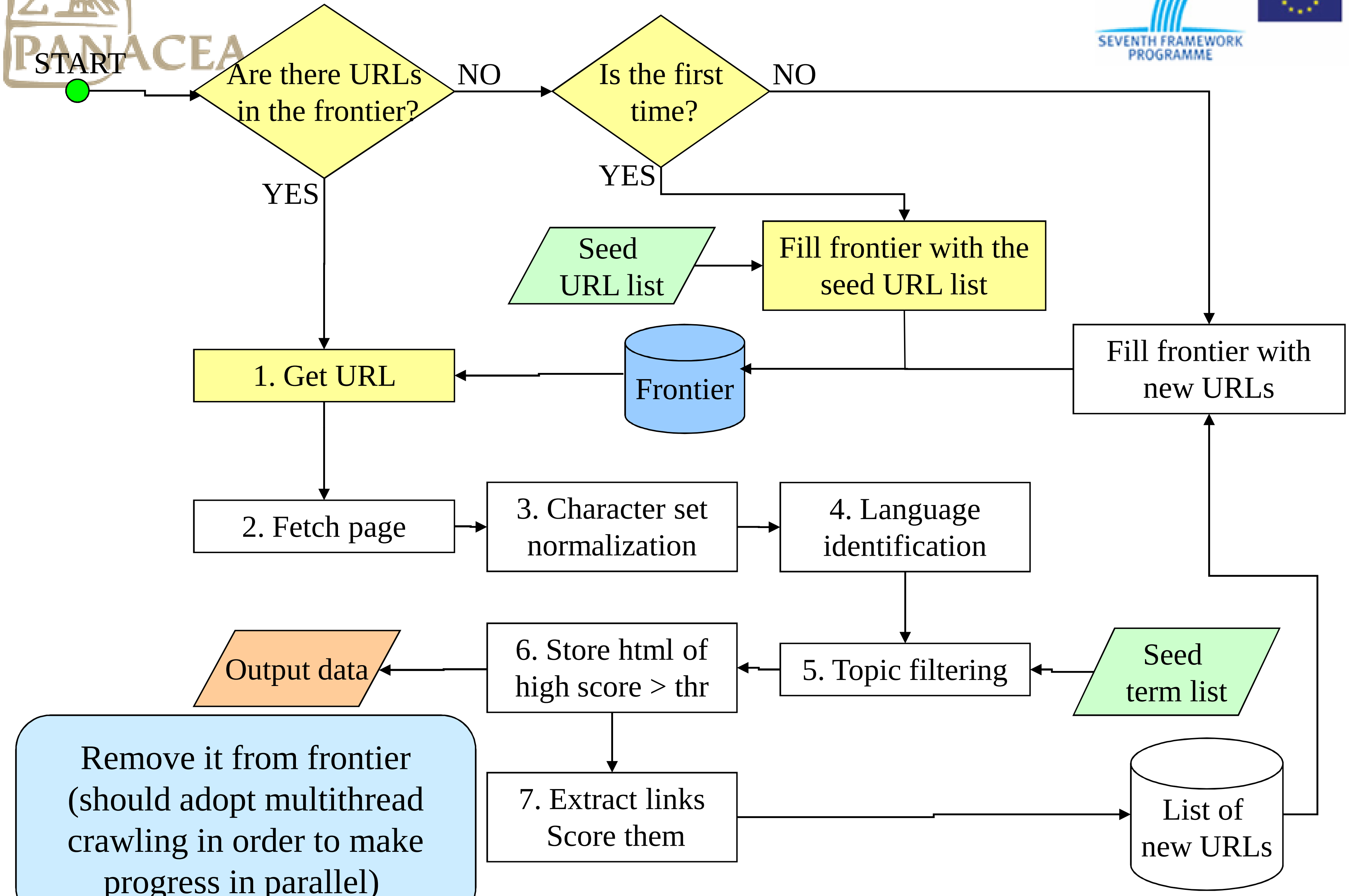
Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
Is the first time?
NO
YES
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
1. Get URL
Frontier
2. Fetch page
3. Character set normalization
4. Language identification
Output data
6. Store html of high score > thr
5. Topic filtering
Seed term list
7. Extract links Score them
List of new URLs

Monolingual data focused crawling (Workflow)

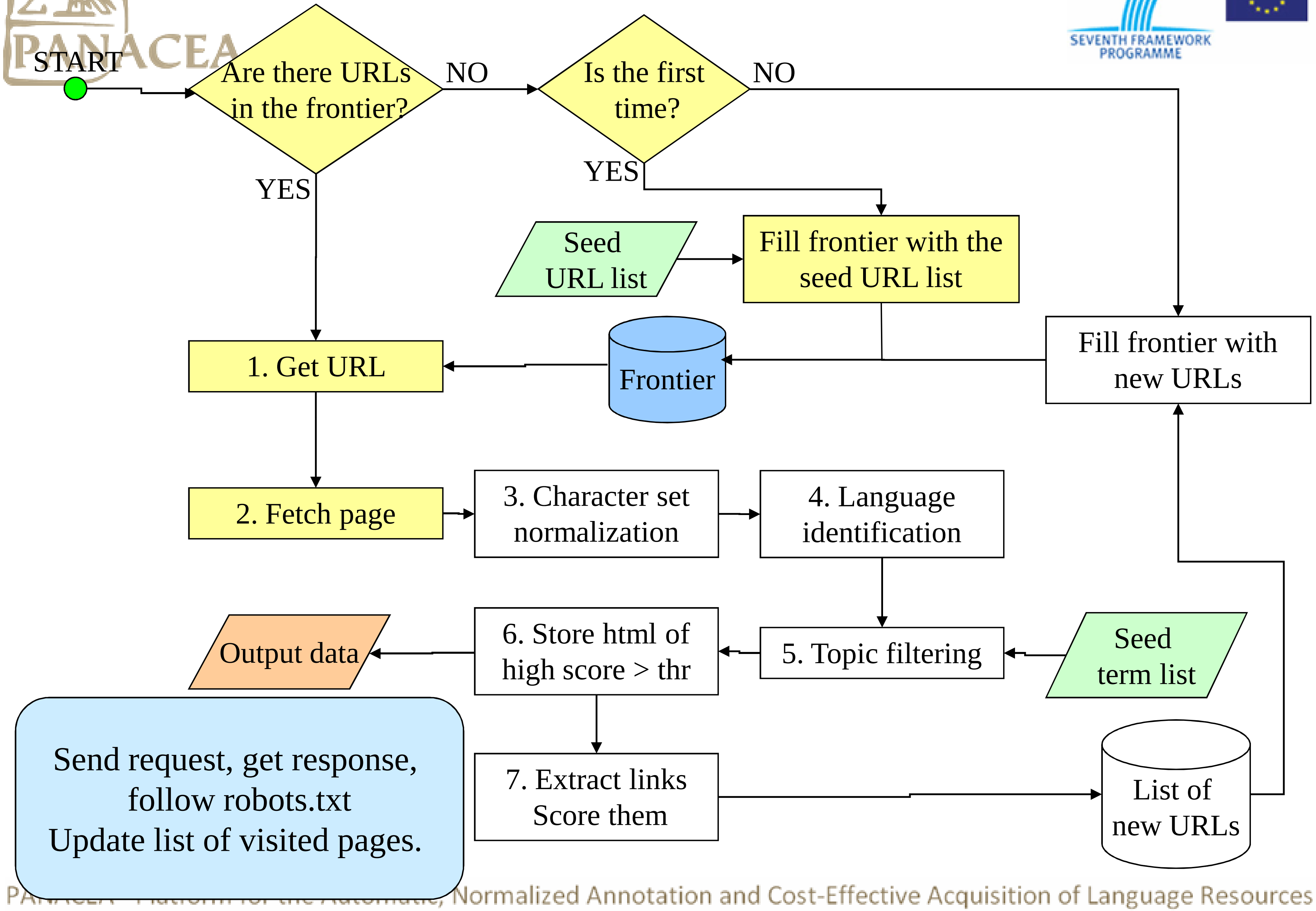

START
Are there URLs in the frontier?
NO
Is the first time?
NO
YES
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
Frontier
1. Get URL
2. Fetch page
3. Character set normalization
4. Language identification
5. Topic filtering
Seed term list
6. Store html of high score > thr
Output data
7. Extract links Score them
List of new URLs

Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
Is the first time?
NO
YES
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
1. Get URL
Frontier
2. Fetch page
3. Character set normalization
4. Language identification
6. Store html of high score > thr
5. Topic filtering
Seed term list
Output data
7. Extract links Score them
List of new URLs

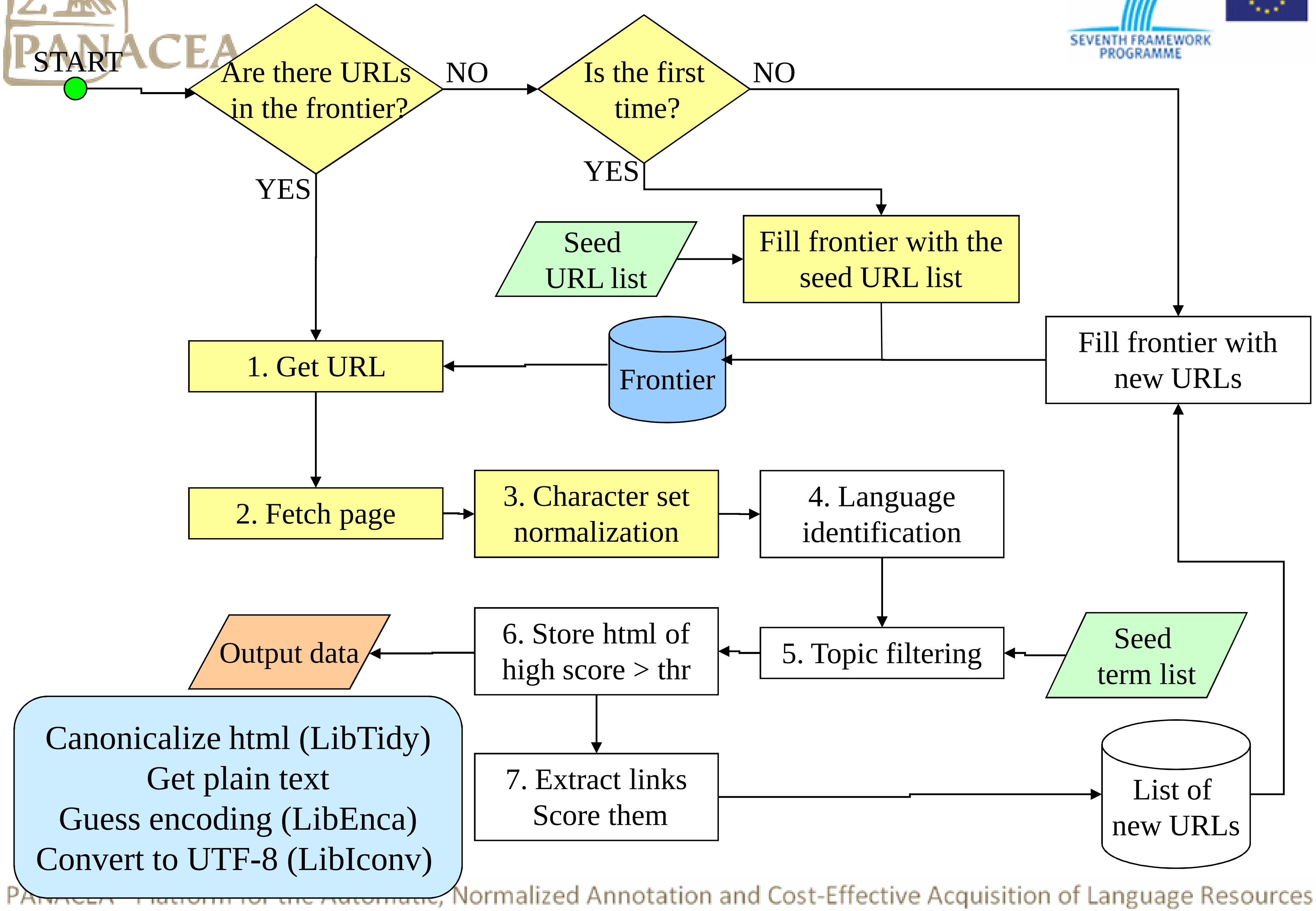
Remove it from frontier (should adopt multithread crawling in order to make progress in parallel)

Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
Is the first time?
NO
YES
YES
Seed URL list
Fill frontier with the seed URL list
Frontier
Fill frontier with new URLs
1. Get URL
2. Fetch page
3. Character set normalization
4. Language identification
5. Topic filtering
Seed term list
6. Store html of high score > thr
Output data
7. Extract links Score them
List of new URLs

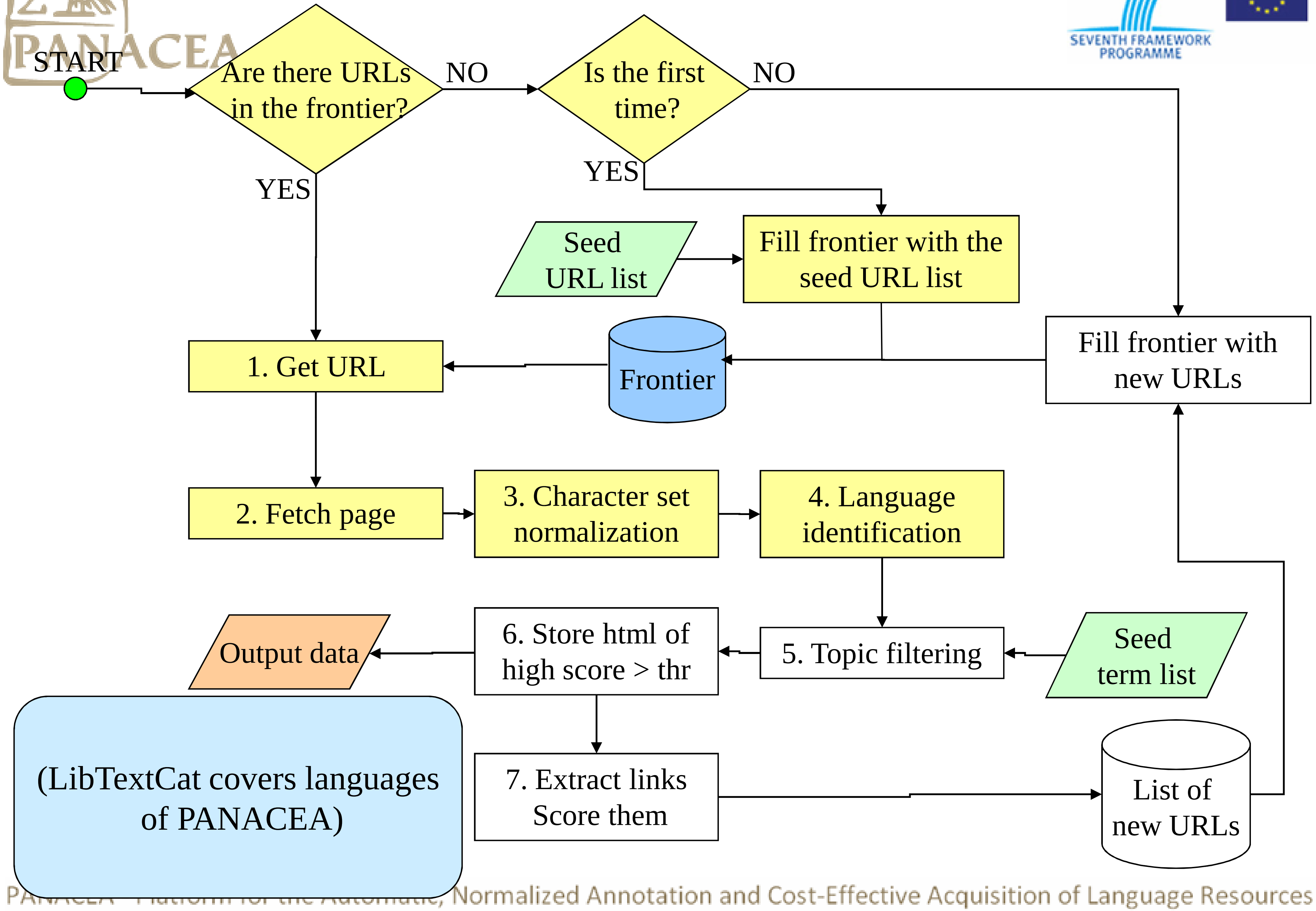
Send request, get response,
follow robots.txt
Update list of visited pages.

Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
Is the first time?
NO
YES
YES
Seed URL list
Fill frontier with the seed URL list
Frontier
Fill frontier with new URLs
1. Get URL
2. Fetch page
3. Character set normalization
4. Language identification
5. Topic filtering
Seed term list
6. Store html of high score > thr
Output data
7. Extract links Score them
List of new URLs

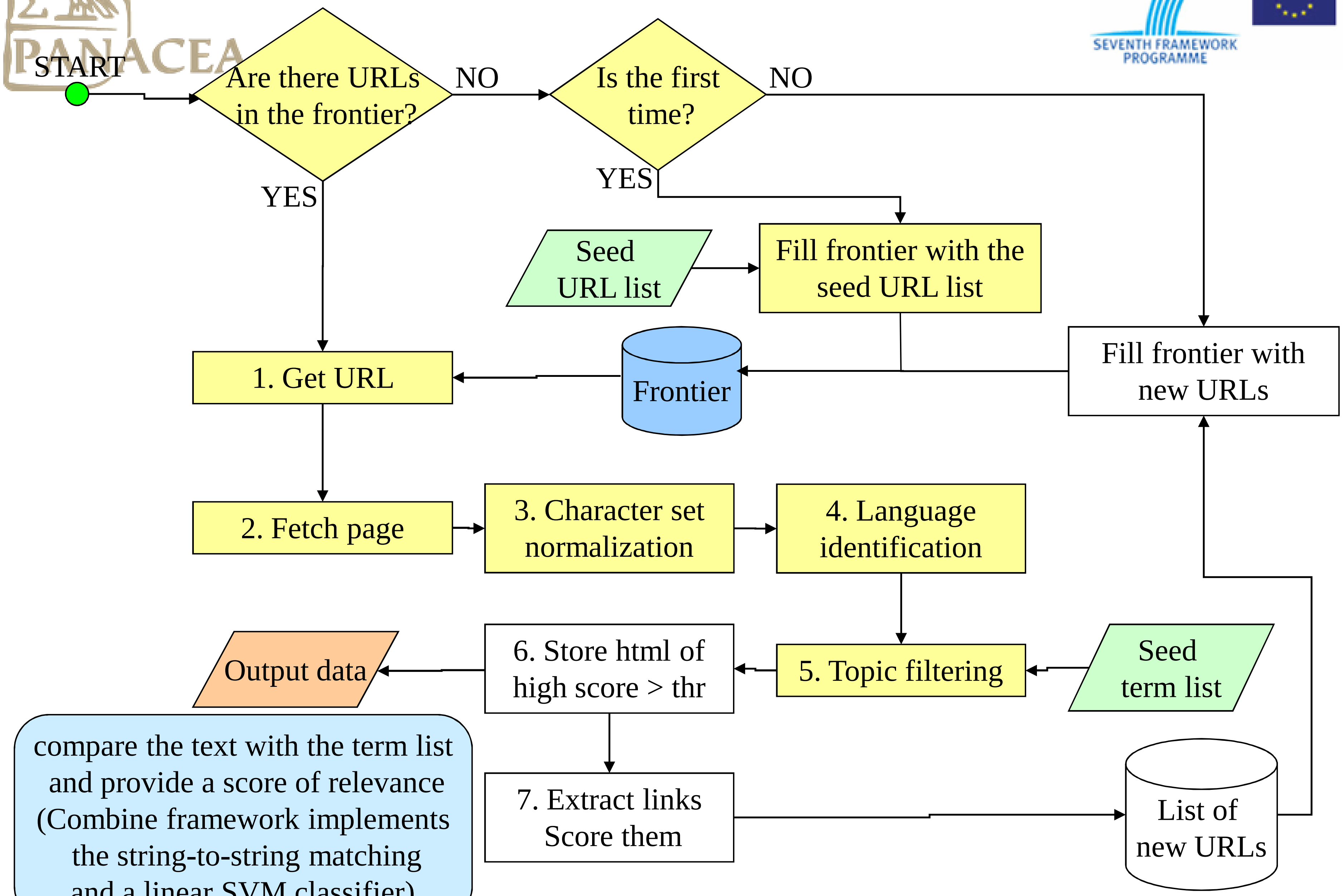
Canonicalize html (LibTidy)
Get plain text
Guess encoding (LibEnca)
Convert to UTF-8 (LibIconv)

Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
YES
Is the first time?
NO
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
1. Get URL
Frontier
2. Fetch page
3. Character set normalization
4. Language identification
Output data
6. Store html of high score > thr
5. Topic filtering
Seed term list
7. Extract links Score them
List of new URLs

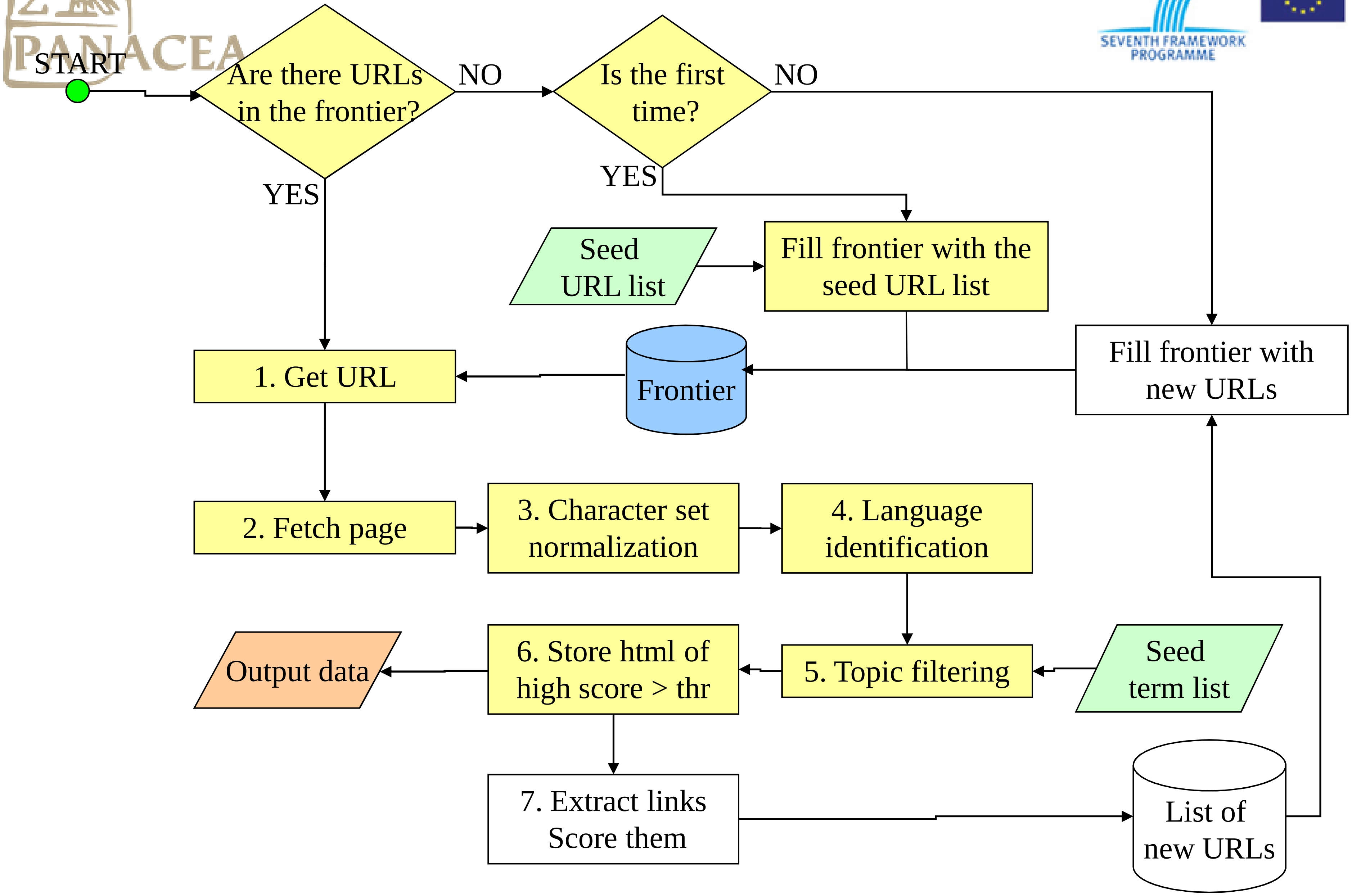
(LibTextCat covers languages of PANACEA)

Monolingual data focused crawling (Workflow)

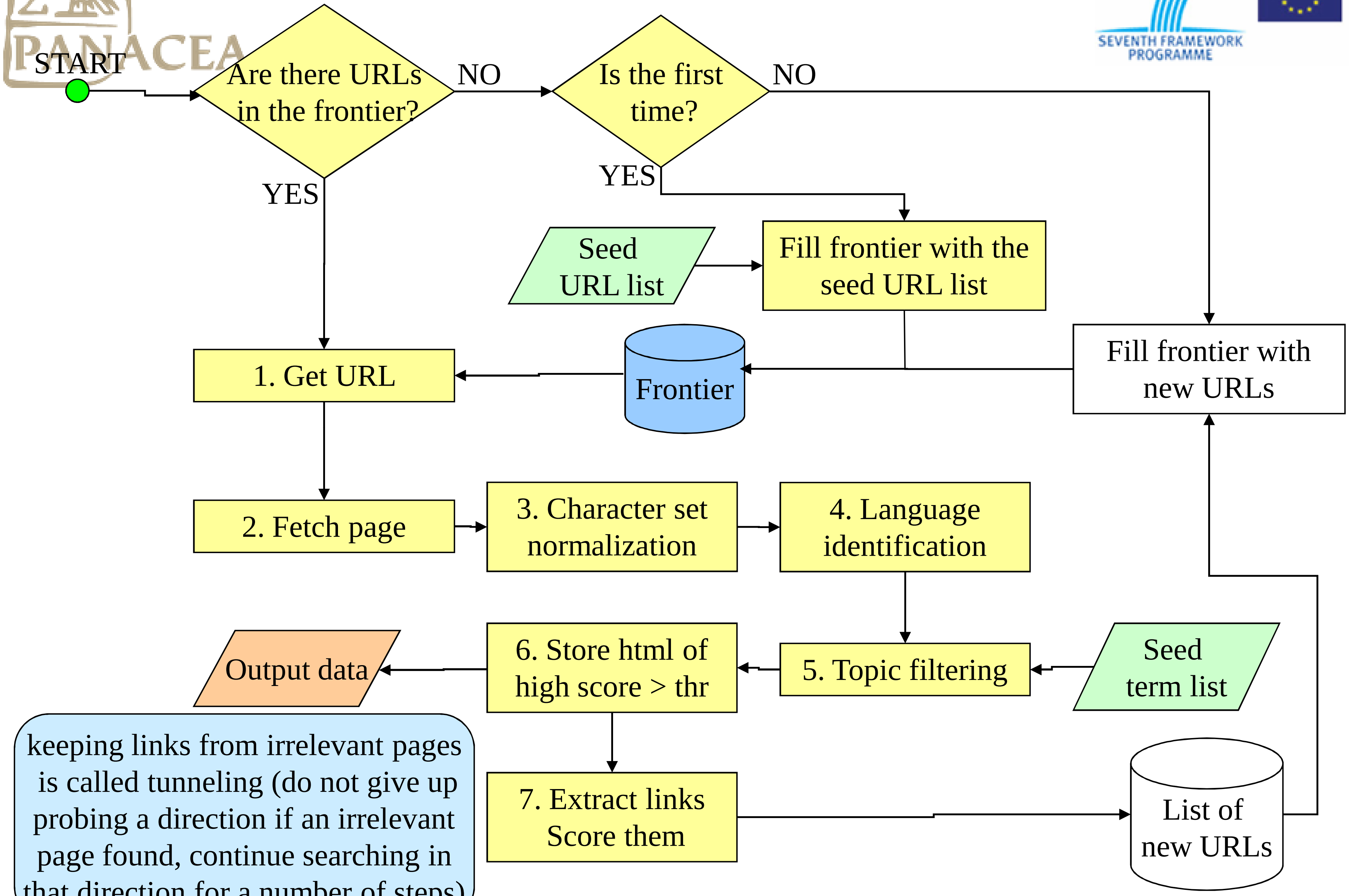

START
Are there URLs in the frontier?
NO
Is the first time?
NO
YES
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
1. Get URL
Frontier
2. Fetch page
3. Character set normalization
4. Language identification
Output data
6. Store html of high score > thr
5. Topic filtering
Seed term list
7. Extract links Score them
List of new URLs

compare the text with the term list and provide a score of relevance (Combine framework implements the string-to-string matching and a linear SVM classifier)

Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
YES
Is the first time?
NO
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
1. Get URL
Frontier
2. Fetch page
3. Character set normalization
4. Language identification
Output data
6. Store html of high score > thr
5. Topic filtering
Seed term list
7. Extract links Score them
List of new URLs

Monolingual data focused crawling (Workflow)



START

Are there URLs in the frontier?

NO

Is the first time?

NO

YES

YES

Seed URL list

Fill frontier with the seed URL list

Frontier

Fill frontier with new URLs

1. Get URL

2. Fetch page

3. Character set normalization

4. Language identification

5. Topic filtering

Seed term list

6. Store html of high score > thr

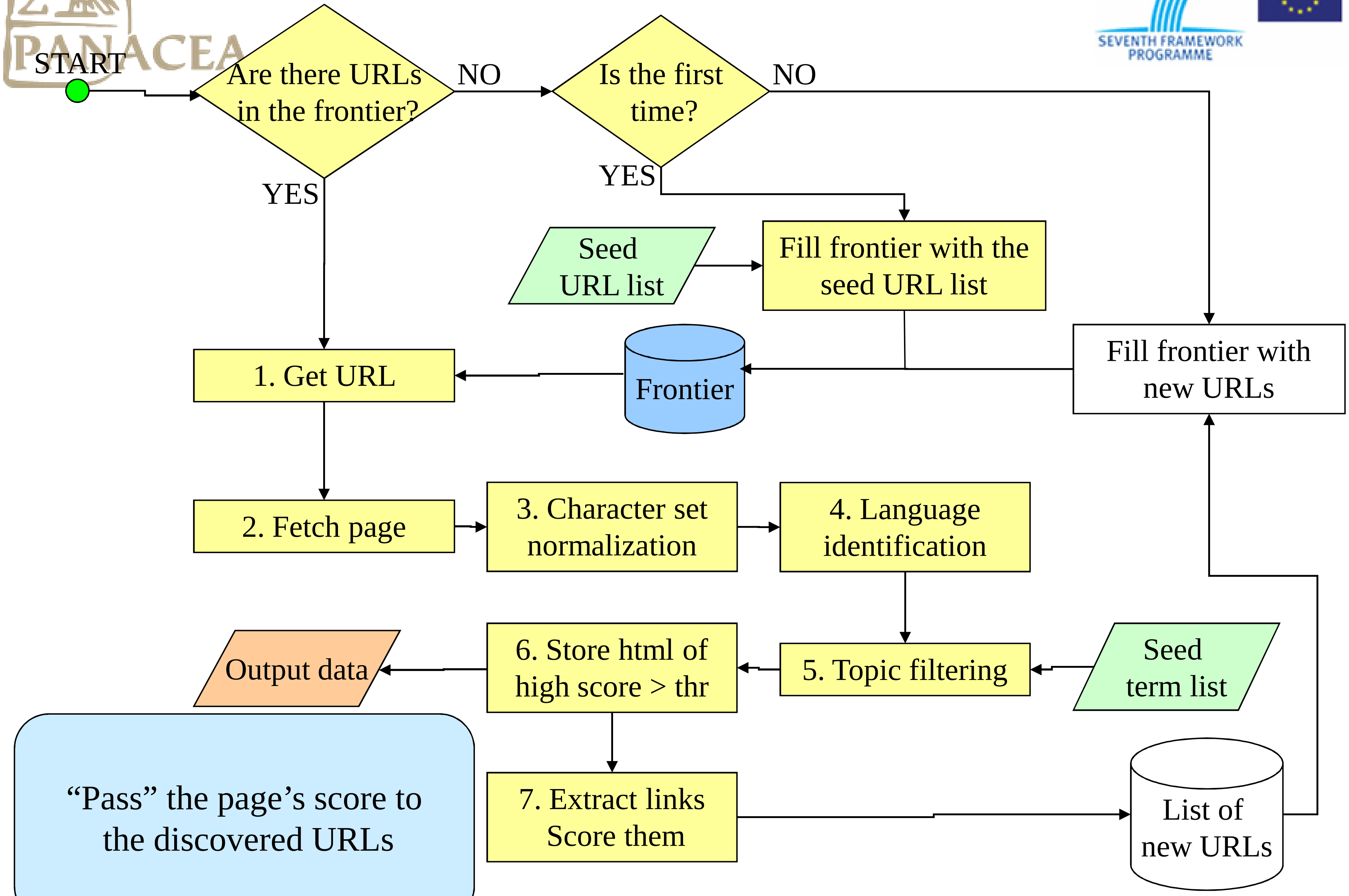
Output data

7. Extract links Score them

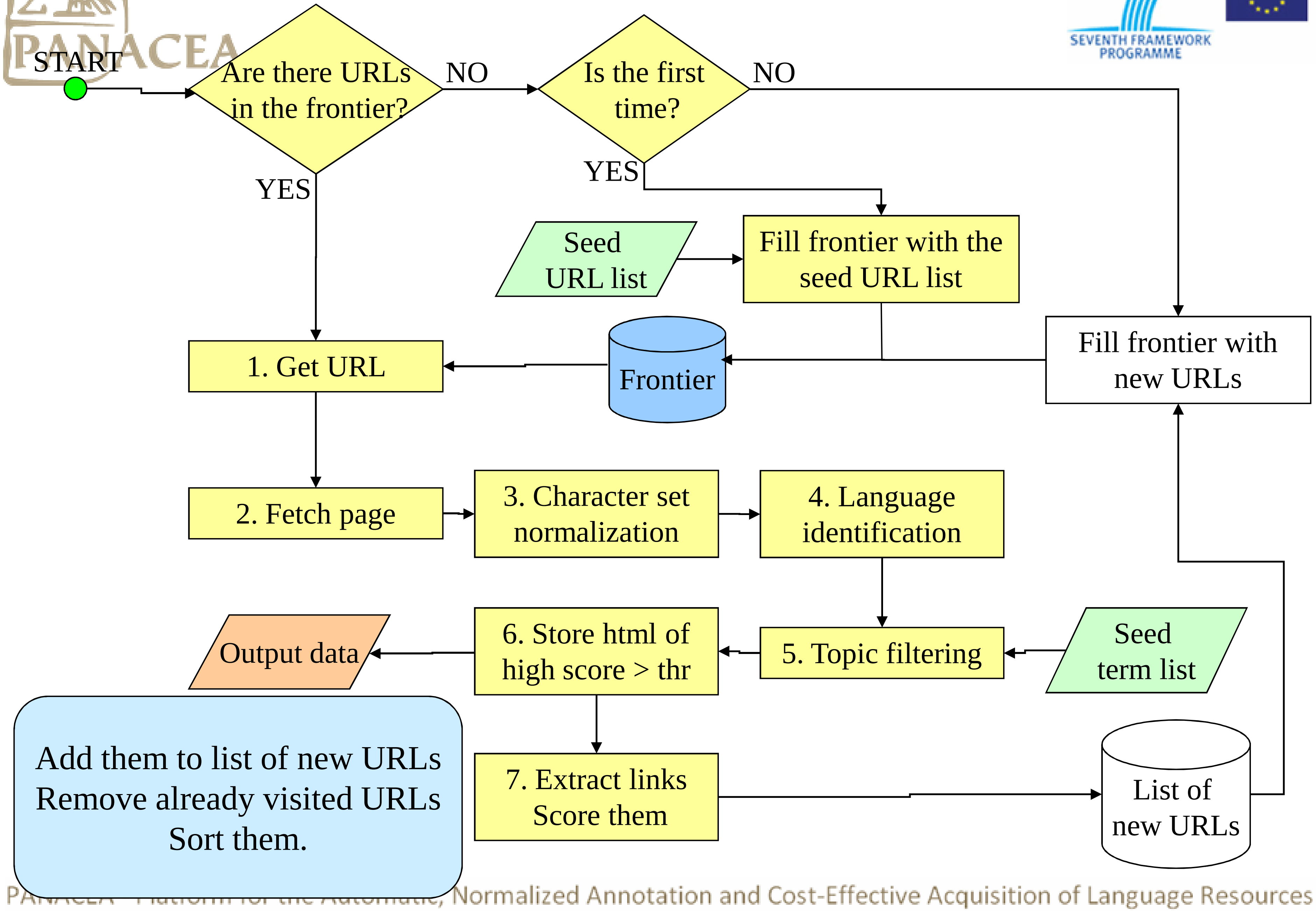
List of new URLs

keeping links from irrelevant pages is called tunneling (do not give up probing a direction if an irrelevant page found, continue searching in that direction for a number of steps)

Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
Is the first time?
NO
YES
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
1. Get URL
Frontier
2. Fetch page
3. Character set normalization
4. Language identification
Output data
6. Store html of high score > thr
5. Topic filtering
Seed term list
“Pass” the page’s score to the discovered URLs
7. Extract links Score them
List of new URLs

Monolingual data focused crawling (Workflow)



START

Are there URLs in the frontier?

NO

Is the first time?

NO

YES

YES

Seed URL list

Fill frontier with the seed URL list

Fill frontier with new URLs

1. Get URL

Frontier

2. Fetch page

3. Character set normalization

4. Language identification

5. Topic filtering

Seed term list

6. Store html of high score > thr

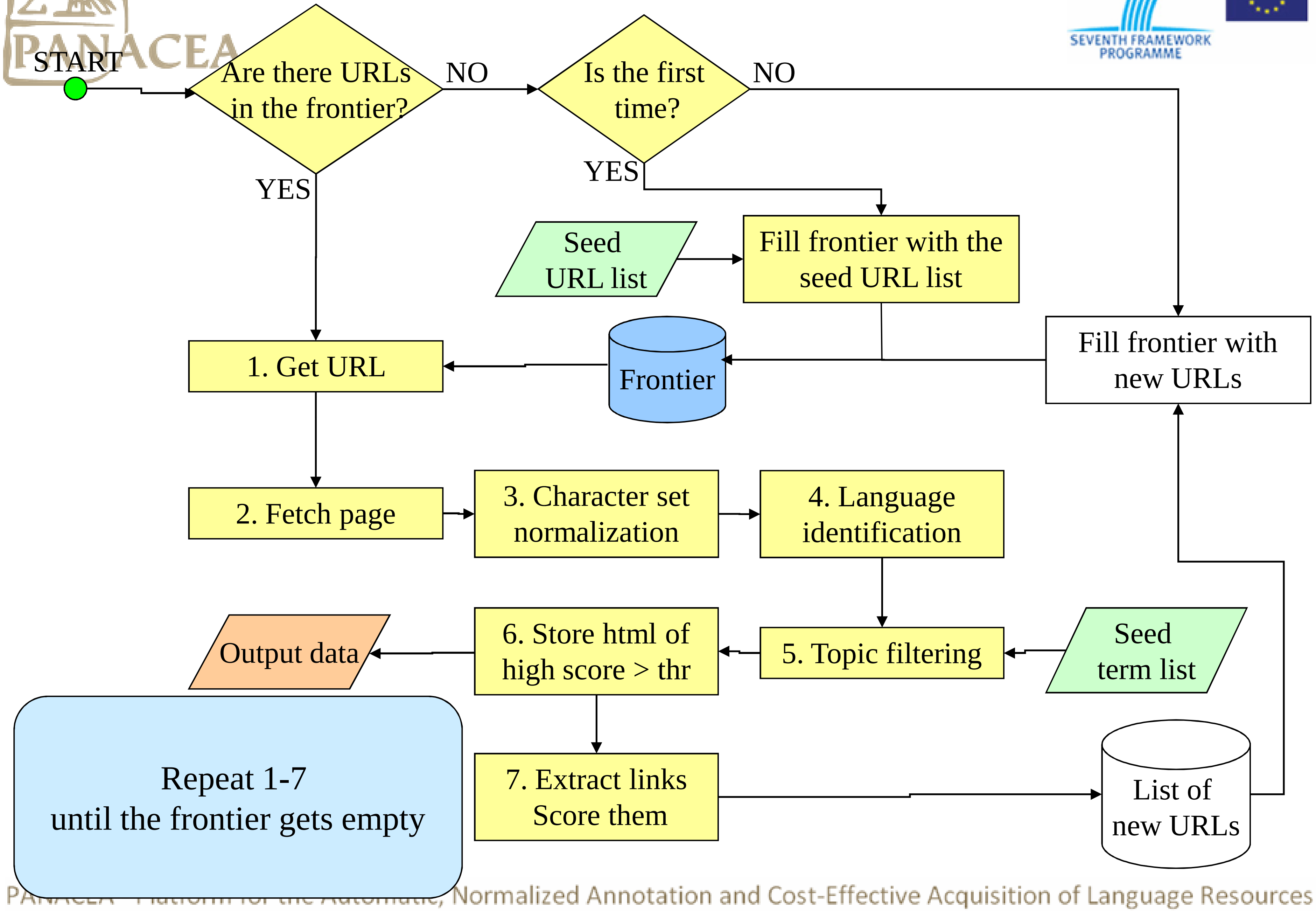
Output data

7. Extract links Score them

List of new URLs

Add them to list of new URLs
Remove already visited URLs
Sort them.

Monolingual data focused crawling (Workflow)



START

Are there URLs in the frontier?

NO

Is the first time?

NO

YES

YES

Seed URL list

Fill frontier with the seed URL list

Frontier

Fill frontier with new URLs

1. Get URL

2. Fetch page

3. Character set normalization

4. Language identification

5. Topic filtering

Seed term list

6. Store html of high score > thr

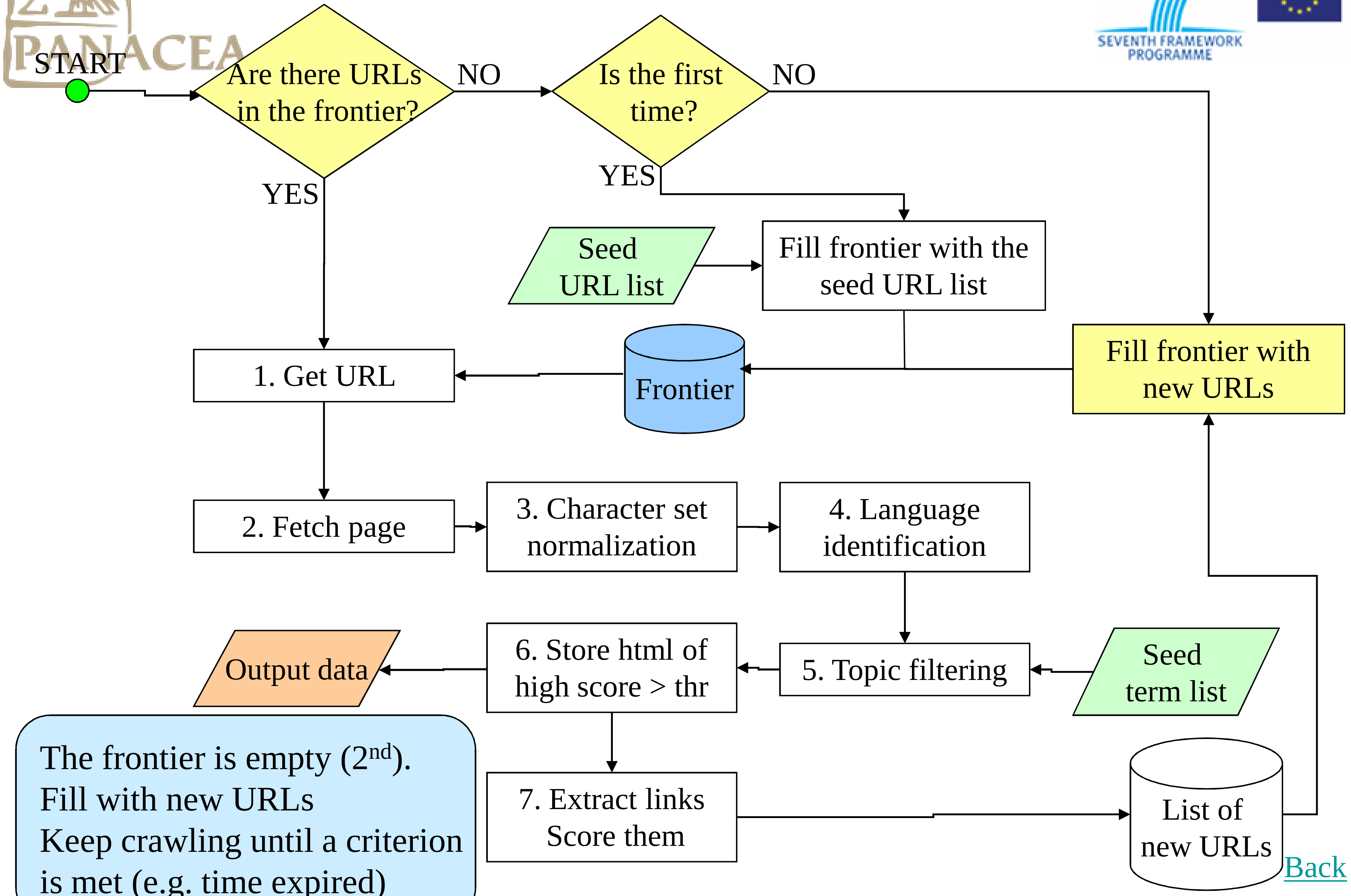
Output data

7. Extract links Score them

List of new URLs

Repeat 1-7
until the frontier gets empty

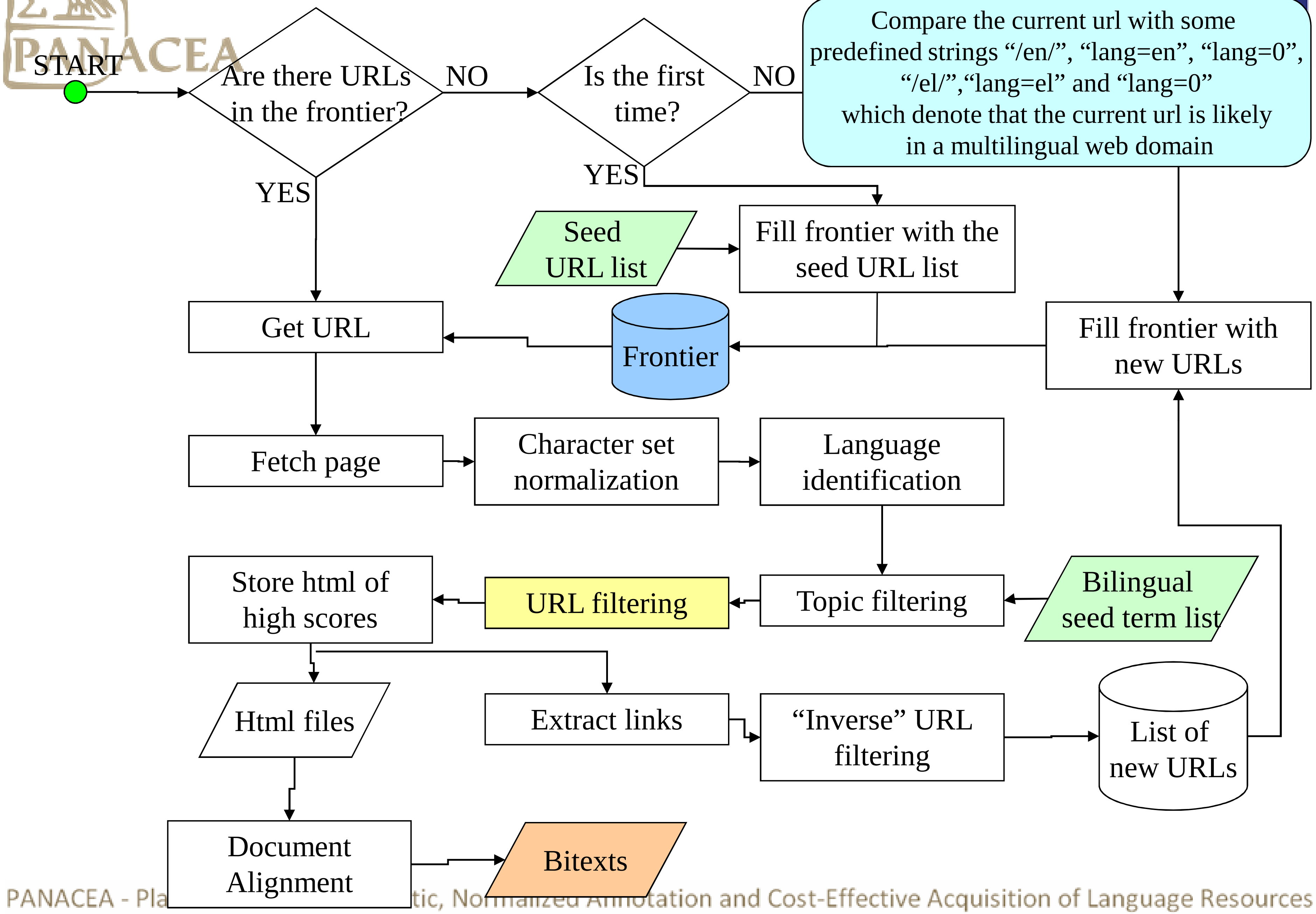
Monolingual data focused crawling (Workflow)


START
Are there URLs in the frontier?
NO
YES
Is the first time?
NO
YES
Seed URL list
Fill frontier with the seed URL list
Fill frontier with new URLs
1. Get URL
Frontier
2. Fetch page
3. Character set normalization
4. Language identification
Output data
6. Store html of high score > thr
5. Topic filtering
Seed term list
7. Extract links Score them
List of new URLs

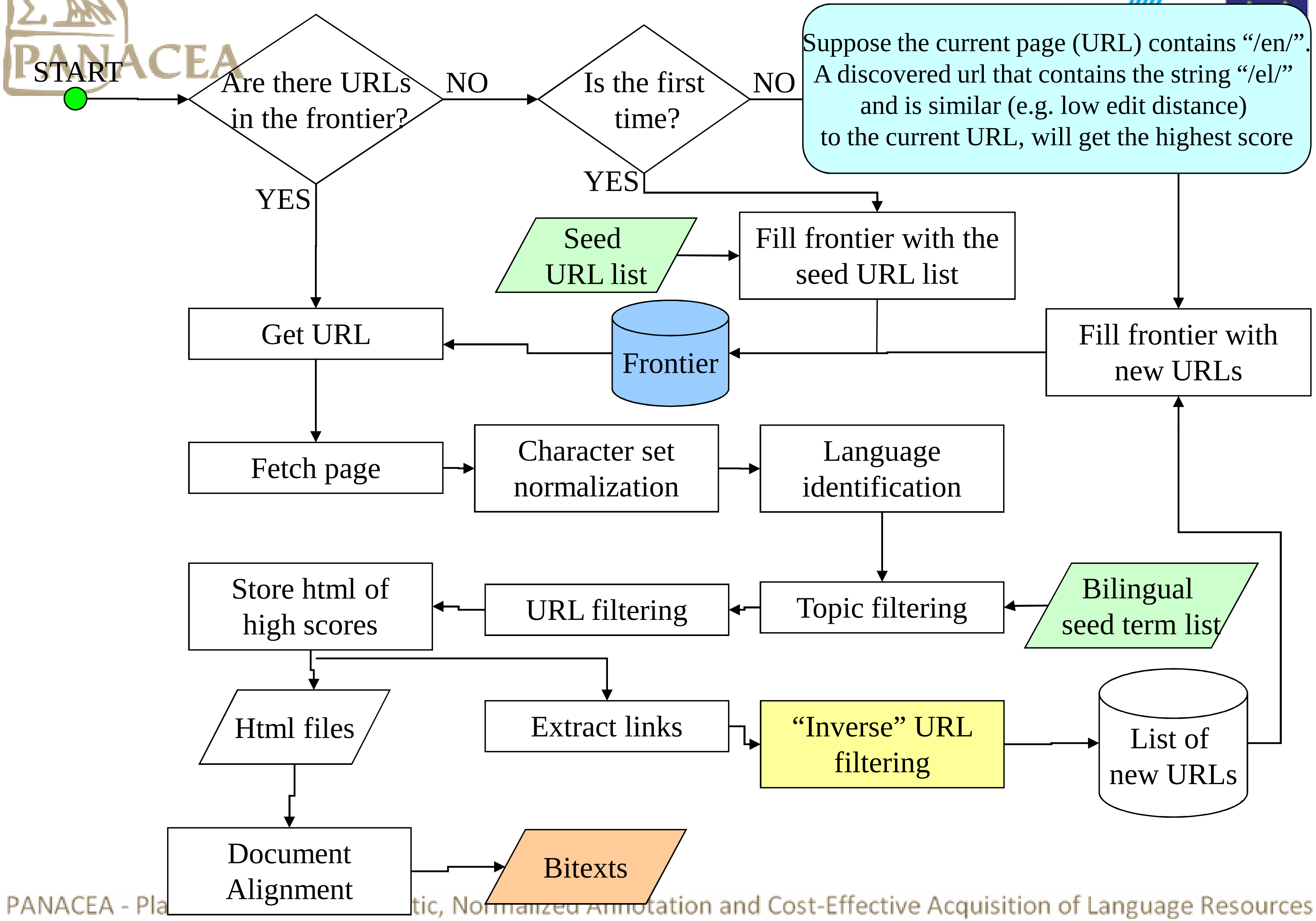
The frontier is empty (2nd).
Fill with new URLs
Keep crawling until a criterion is met (e.g. time expired)

Back

Bilingual data focused crawling

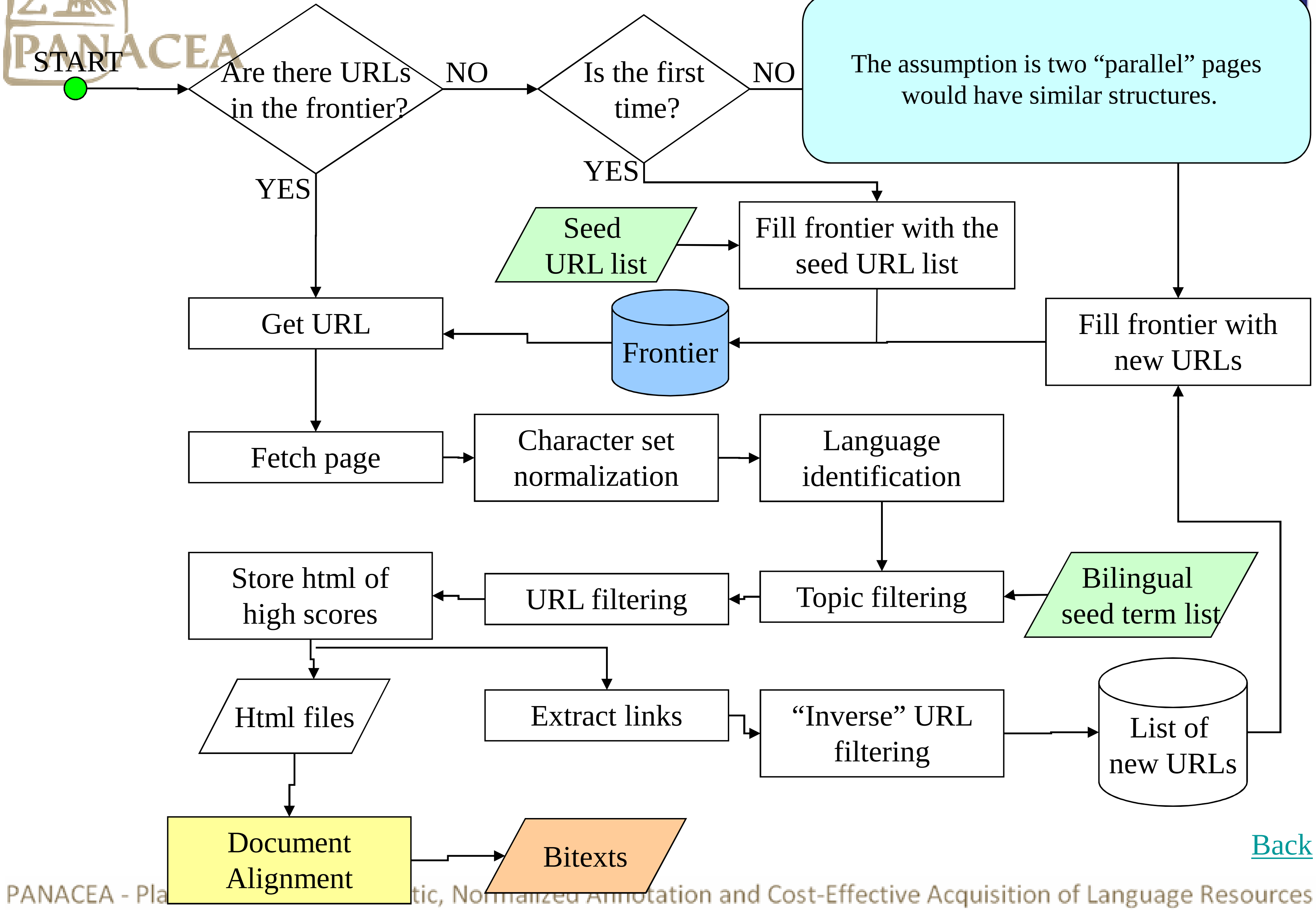

START
Are there URLs in the frontier?
NO
Is the first time?
NO
Compare the current url with some predefined strings "/en/", "lang=en", "lang=0", "/el/","lang=el" and "lang=0" which denote that the current url is likely in a multilingual web domain
YES
YES
Seed URL list
Fill frontier with the seed URL list
Get URL
Frontier
Fill frontier with new URLs
Fetch page
Character set normalization
Language identification
Store html of high scores
URL filtering
Topic filtering
Bilingual seed term list
Html files
Extract links
"Inverse" URL filtering
List of new URLs
Document Alignment
Bitexts

Bilingual data focused crawling



Suppose the current page (URL) contains "/en/".
A discovered url that contains the string "/el/"
and is similar (e.g. low edit distance)
to the current URL, will get the highest score

Bilingual data focused crawling



The assumption is two "parallel" pages would have similar structures.

Back