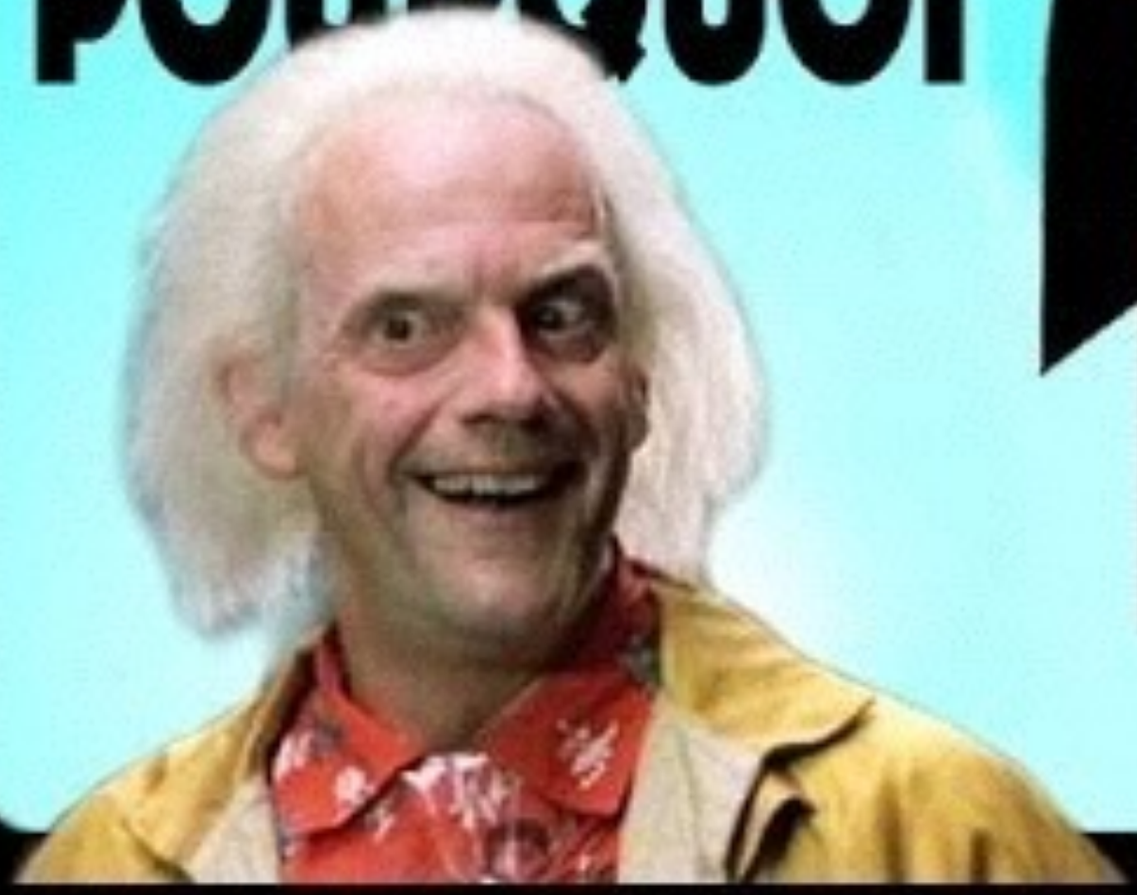
common search:

Et si on recodait Google en Python ?

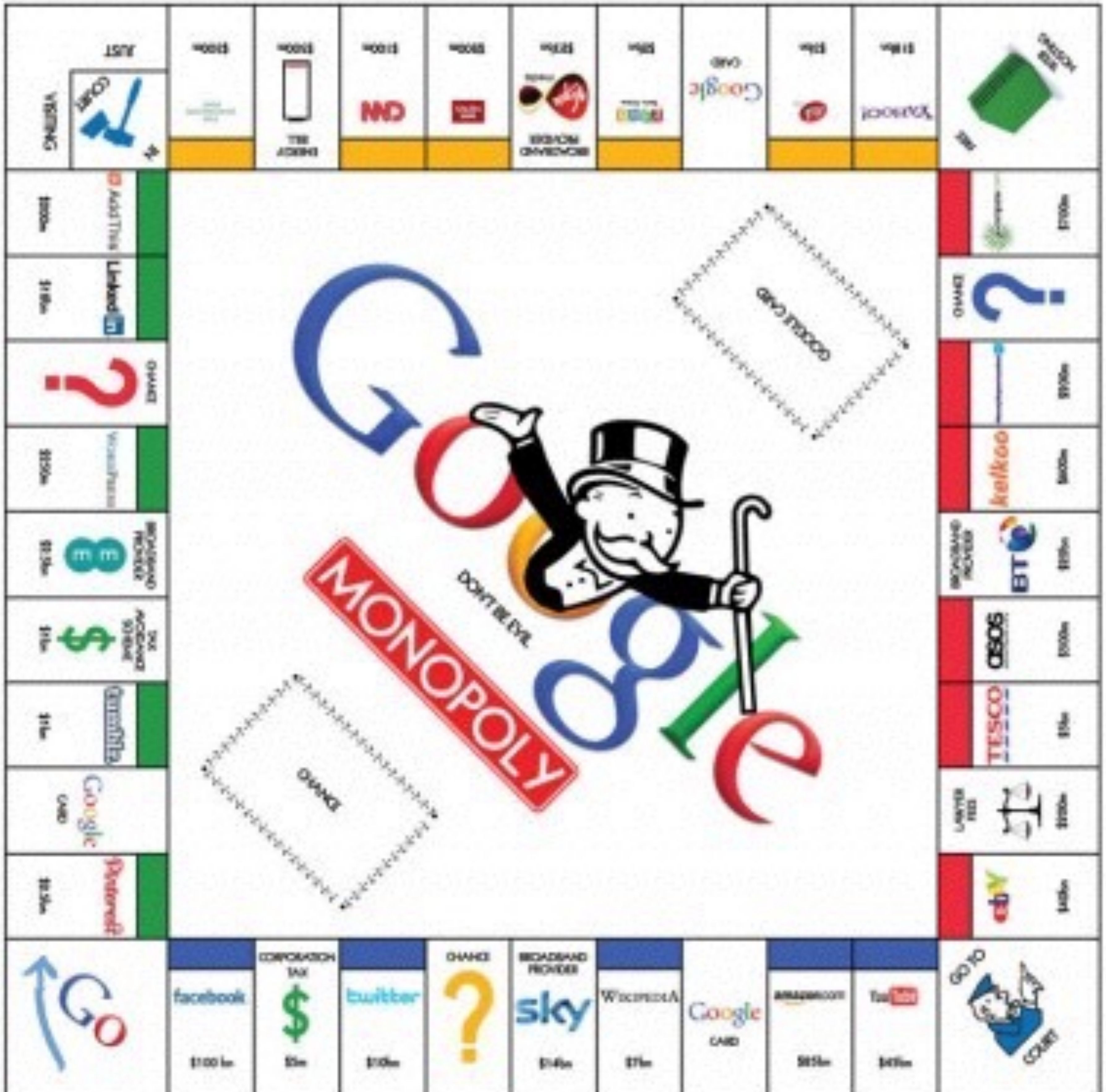
PyCon-FR 2016

sylvain@sylvainzimmer.com

MAIS POURQUOI




FUN
ZONE



Go
facebook
CORPORATION TAX
twitter
CHANCE
BROADBAND PROVIDER
sky
WIKIPEDIA
Google
CARD
amazon.com
YouTube
GO TO
COURT
Google
MONOPOLY
DON'T BE EVIL
CHANCE
GOOGLE CARD
kelkoo
BT
asos
TESCO
ebay
LinkedIn
WordPress
Pinterest
CNN
Yahoo!

transparence

reproductibilité

common search:

common search: python EN

About 41380 results

Welcome to **Python**.org
www.python.org
The official home of the **Python** Programming Language

Dive Into **Python**
www.diveintopython.net
This book lives at . If you're reading it somewhere else, you may not have the latest version.

The Eric **Python** IDE
eric-ide.python-projects.org
Eric is a full featured **Python** editor and IDE, written in **Python**. It is based on the cross platform Qt gui toolkit, integrating the highly flexible Scintilla...

Starship
www.python.net
The home of pythonistas

Tutorials, **Python** Courses: Online and On Site
python-course.eu
Free comprehensive online tutorials suitable for self-study and high-quality on site **Python** courses in Europe, Canada (Toronto) and the US

learning **python** | one man's journey into **python**…
www.learningpython.com
one man's journey into **python**...

https://uidemo.commonsearch.org

Common Search Explainer | Search test | URL test | About

python | OK | EN

Results (50)

Welcome to Python.org

[debug] https://www.python.org/

The official home of the Python Programming Language

docid -4478921722574158000

static rank 0.7923434

ES score 87.821815

ES explain

```
47.75143 | sum of:
    47.75143 | function score, product of:
        60.87483 | max plus 0.5 times others of:
            60.768433 | weight(domain_words:python in 77874) [PerFieldSimilarity], result of:
                60.768433 | score(doc=77874,freq=1.0 = termFreq=1.0
), product of:
                    8.0 | boost
                    5.97652 | idf(docFreq=9023, maxDocs=3555860)
                    1.2709827 | tfNorm, computed from:
                        1.0 | termFreq=1.0
                        1.0 | parameter k1
                        0.75 | parameter b
                        2.31778 | avgFieldLength
                        1.0 | fieldLength
            0.21279304 | weight(body:python in 77874) [PerFieldSimilarity], result of:
                0.21279304 | score(doc=77874,freq=21.0), product of:
                    0.14198984 | queryWeight, product of:
                        6.976686 | idf(docFreq=9021, maxDocs=3555860)
                        0.020352047 | queryNorm
                    1.4986497 | fieldWeight in 77874, product of:
                        4.582576 | tf(freq=21.0), with freq of:
                            21.0 | termFreq=21.0
                        6.976686 | idf(docFreq=9021, maxDocs=3555860)
                        0.046875 | fieldNorm(doc=77874)
        0.78441995 | min of:
            0.78441995 | function score, score mode [multiply]
                0.7923434 | function score, product of:
                    1.0 | match filter: *:*
```

https://explain.commonsearch.org/?q=python&g=en

Google's early Python code

Python (1.2 IIRC) would occasionally just core dump while running the crawler. It was completely stock, no C++ modules compiled in or dynamically linked, just bog standard.

[...] no unit tests, and its "system tests" were minimal at best, absent at worst.

[...] there was originally some controversy about the switch. However, when the new C++ system was turned on and used fewer machines to crawl 5x faster with higher reliability, the practical question was settled.

Python was "abandoned" from the core search stack around 2000.

https://www.quora.com/Why-did-Google-move-from-Python-to-C++-for-use-in-its-crawler

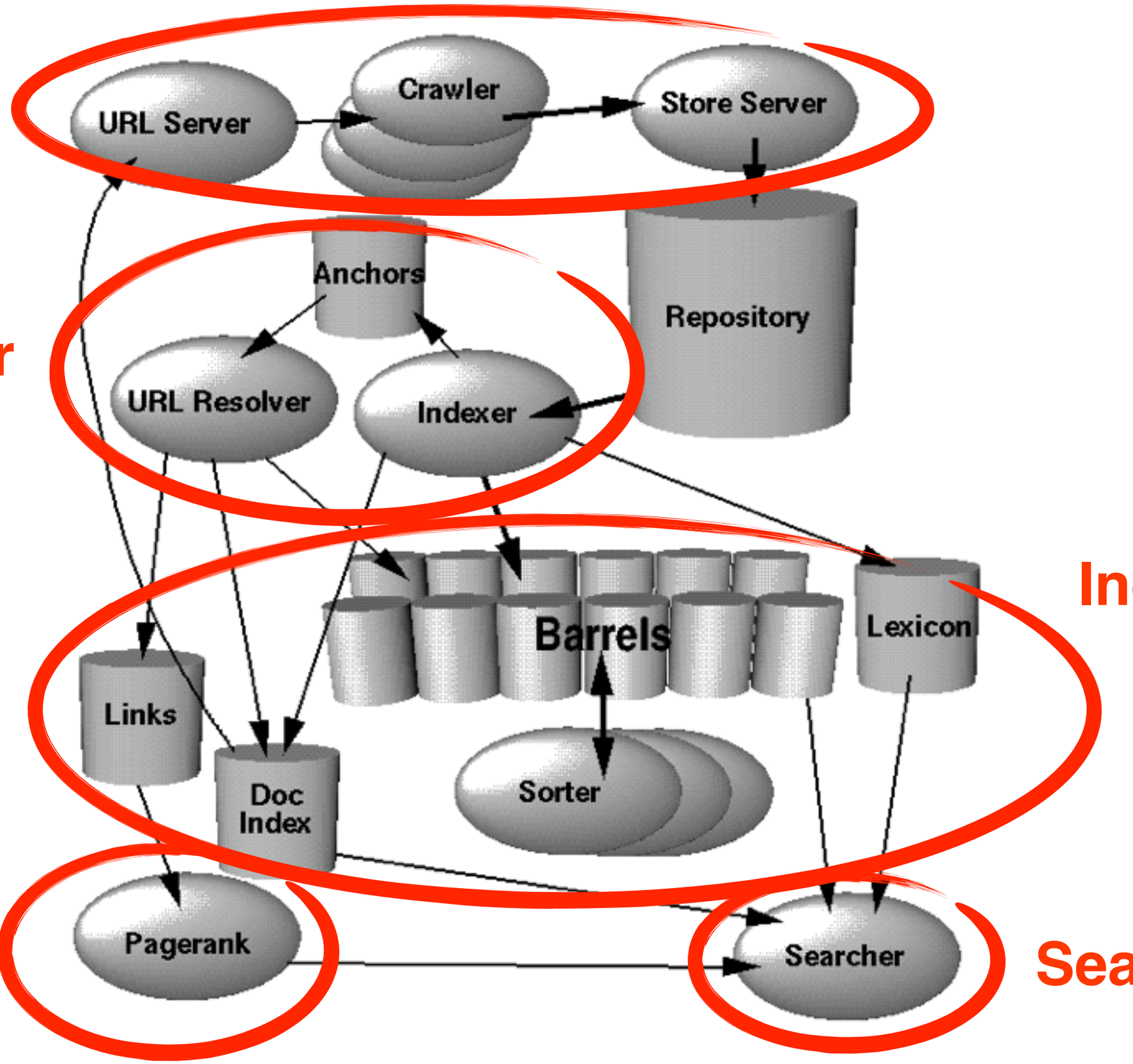
Qu'est-ce qui a changé depuis ?

- Stabilité & écosystème
- Librairies performantes en C / Cython
- Evolution des bottlenecks
- PyPy?


SEARCH ENGINES
HOW DO THEY WORK?

Crawler

Parser



Index

Ranker

Searcher

The Anatomy of a Large-Scale Hypertextual Web Search Engine (1998)

http://infolab.stanford.edu/~backrub/google.html

Crawler

Scrapy

An open source and collaborative framework
for extracting the data you need from websites.
In a fast, simple, yet extensible way.

Install the latest version of Scrapy

Scrapy 1.1

```
$ pip install scrapy
```

PyPI Conda APT Source

pypi v1.1.3 wheel yes Python 3 Porting Status coverage 83%

Build and run your **web spiders**

Terminal

```
$ pip install scrapy
$ cat > myspider.py <<EOF
import scrapy

class BlogSpider(scrapy.Spider):
    name = 'blogspider'
    start_urls = ['https://blog.scrapinghub.com']

    def parse(self, response):
        for title in response.css('h2.entry-title'):
            yield {'title': title.css('a ::text').extract_first()}

        next_page = response.css('div.prev-post > a ::attr(href)').extract_first()
        if next_page:
            yield scrapy.Request(response.urljoin(next_page), callback=self.parse)
EOF
$ scrapy runspider myspider.py
```

http://scrapy.org

cocrawler / cocrawler

Watch 3 | Unstar 6 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs

CoCrawler is a versatile web crawler built using modern tools and concurrency.

193 commits | 1 branch | 0 releases | 1 contributor | Apache-2.0

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

wumpus process affinity is a nice win — Latest commit c87b825 5 days ago

cocrawler	process affinity is a nice win	5 days ago
examples	bump versions; measure decode cpu burn; tweaks	2 months ago
tests	process affinity is a nice win	5 days ago
.gitignore	shinier	3 months ago
.travis.yml	ok 3.5.0 is gone	29 days ago
LICENSE	initial import	3 months ago
README.md	update blather	6 days ago
TODO	move dns to separate file; first pytest-asyncio tests	12 days ago
requirements.txt	add histograms	7 days ago

http://github.com/cocrawler/cocrawler

HERITRIX

Apache nutch

STORMCRAWLER

August 2016 Crawl Archive Now Available

September 16, 2016 **Sebastian Nagel**

The crawl archive for August 2016 is now available! The archive located in the **commoncrawl** bucket at **crawl-data/CC-MAIN-2016-36/** contains more than 1.61 billion web pages.

To extend the seed list, we've added 50 million hosts from the Common Search host-level pagerank data set. While many of these hosts may already be known, and some may not provide crawlable content, the number of crawled hosts has grown by 18 million (or 50%) and there are 8 million more unique domains (plus 35%).

Together with the August 2016 crawl archive we also release data sets containing robots.txt files and responses without content (404s, redirects, etc.). More information can be found in a separate blog post.

To assist with exploring and using the dataset, we provide gzipped files that list:

- all segments (CC-MAIN-2016-36/segment.paths.gz)
- all WARC files (CC-MAIN-2016-36/warc.paths.gz)
- all WAT files (CC-MAIN-2016-36/wat.paths.gz)
- all WET files (CC-MAIN-2016-36/wet.paths.gz)

By simply adding either **s3://commoncrawl/** or **https://commoncrawl.s3.amazonaws.com/** to each

Recent Posts

August 2016 Crawl Archive Now Available

Data Sets Containing Robots.txt Files and Non-200 Responses

July 2016 Crawl Archive Now Available

June 2016 Crawl Archive Now Available

May 2016 Crawl Archive Now Available

http://commoncrawl.org

Parser

HTML parsers

- BeautifulSoup & derivés.
- lxml
- html5lib
- Gumbo!

google / gumbo-parser

Watch 321 | Star 3,818 | Fork 502

Code | Issues 30 | Pull requests 6 | Projects 0 | Wiki | Pulse | Graphs

An HTML5 parsing library in pure C99

403 commits | 4 branches | 7 releases | 27 contributors | Apache-2.0

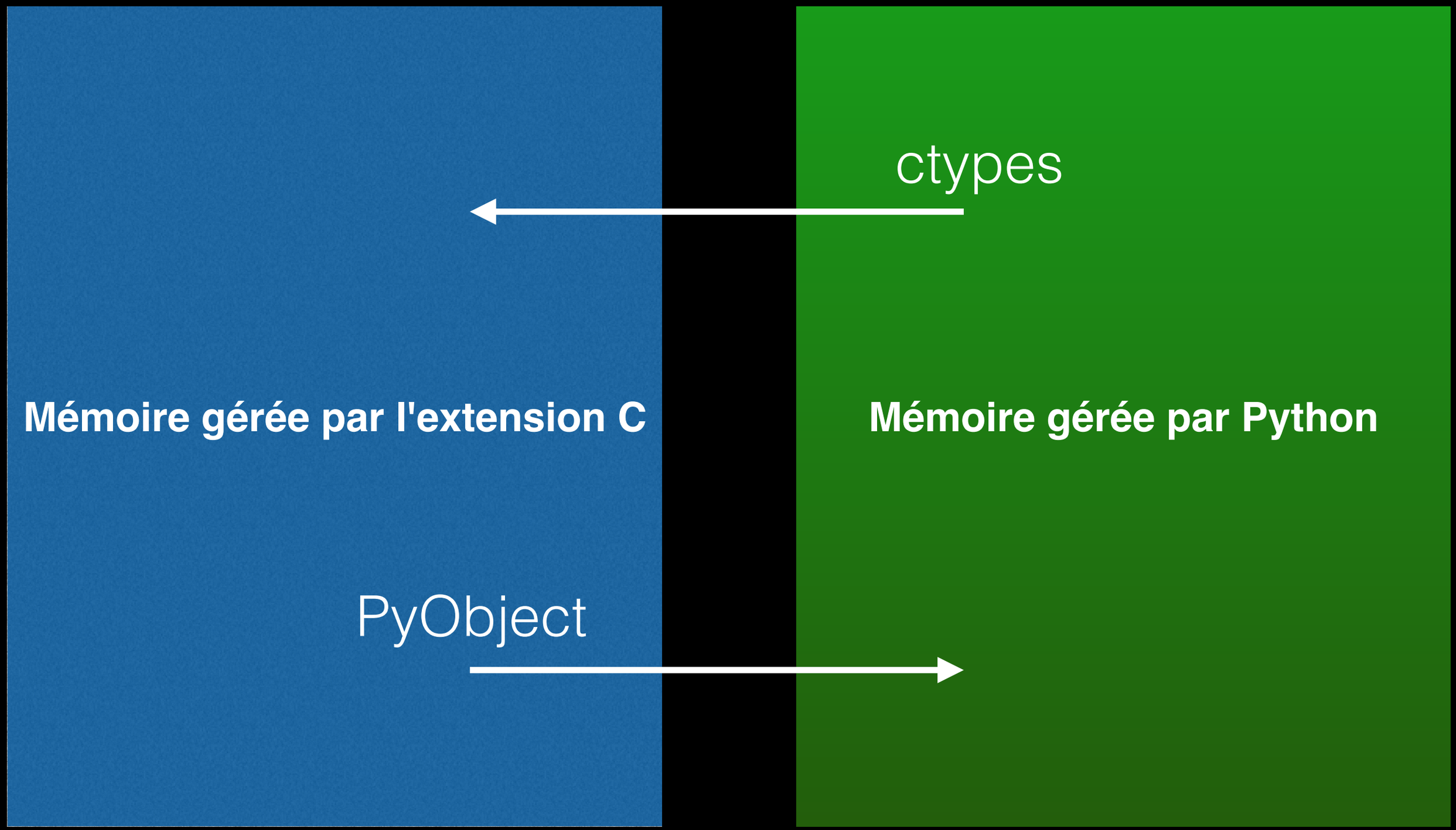
Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

nostrademons committed on GitHub Merge pull request #367 from mominul/patch-1 ... Latest commit aa91b27 on Jun 29

benchmarks	Add baidu benchmark which has been left out of the git repository all...	2 years ago
examples	Recognize templates in serialize and prettyprint	2 years ago
python/gumbo	Update gen_tags.py to exempt generated files from clang-format, and r...	a year ago
src	Fix error mesage use of return value form vnsprintf	10 months ago
testdata @ e633ddf	Move html5lib-tests submodule ref up to include the ruby fix.	2 years ago
tests	Added a test for fragments with multiple nodes.	a year ago
third_party	Integrate gumbo_parser with gtest.	3 years ago
visualc	Update strings.h	2 years ago
.clang-format	Reformat the source code with clang-format, and add a config file for...	a year ago

https://github.com/google/gumbo-parser

Extensions C en Python



```
class Element(ctypes.Structure):
  _fields_ = [
      ('children', NodeVector),
      ('tag', Tag),
      ('tag_namespace', Namespace),
      ('original_tag', StringPiece),
      ('original_end_tag', StringPiece),
      ('start_pos', SourcePosition),
      ('end_pos', SourcePosition),
      ('attributes', AttributeVector),
      ]

  @property
def tag_name(self):
  original_tag = StringPiece.from_buffer_copy(self.original_tag)
  _tag_from_original_text(ctypes.byref(original_tag))
  if self.tag_namespace == Namespace.SVG:
    svg_tagname = _normalize_svg_tagname(ctypes.byref(original_tag))
    if svg_tagname is not None:
      return str(svg_tagname)
  if self.tag == Tag.UNKNOWN:
    if original_tag.data is None:
      return ''
    return str(original_tag).lower()
  return _tagname(self.tag)
```

Cython!

- Faire le gros du travail en C
- Eviter la conversion de données au maximum
- Générer une extension C pour Python facilement

```
+1: def func():
+2:     a = 0
+3:     for i in xrange(0, 100000000):
+4:         a += i
+5:     return a
 6:
+7: print func()
```

```
 01:
+02: cdef long func():
 03:     cdef long a
 04:     cdef long i
+05:     a = 0
+06:     for i in xrange(0, 100000000):
+07:         a += i
+08:     return a
 09:
+10: res = func()
+11: print res
```

```
 01:
+02: cdef long func():
 03:     cdef long a
 04:     cdef long i
+05:     a = 0
+06:     for i in xrange(0, 100000000):
    for (__pyx_t_1 = 0; __pyx_t_1 < 0x5F5E100; __pyx_t_1+=1) {
      __pyx_v_i = __pyx_t_1;
+07:         a += i
+08:     return a
 09:
+10: res = func()
+11: print res
    __pyx_t_1 = __Pyx_GetModuleGlobalName(__pyx_n_s_res); if (unlikely(!__
    __Pyx_GOTREF(__pyx_t_1);
    if (__Pyx_PrintOne(0, __pyx_t_1) < 0) {__pyx_filename = __pyx_f[0]; __
    __Pyx_DECREF(__pyx_t_1); __pyx_t_1 = 0;
```

https://github.com/sylvinus/cython-simple-examples

Gumbocy

- HTML envoyé au C en UTF-8, sans conversion
- Parcours de l'arbre en Cython
- Gestion de la visibilité & du boilerplate
- Attributs & tags ignorables, ...

https://github.com/commonsearch/gumbocy

urlparse4

`urlparse4` is a performance-focused replacement for Python's `urlparse` module, using C++ code from Chromium's own URL parser.

It is not production-ready yet.

Many credits go to gurl-cython for inspiration.

Differences with Python's `urlparse`

`urlparse4` should be a transparent, drop-in replacement in almost all cases. Still, there are a few differences to be aware of:

- `urlparse4` is 2-7x faster for most operations (see benchmarks below)
- `urlparse4` currently doesn't pass CPython's `test_urlparse.py` suite due to edge cases that Chromium's parser manages differently (usually in accordance to the RFCs, which `urlparse` doesn't follow entirely).
- `urlparse4` only supports Python 2.7 for now

How to install

```
pip install urlparse4
```

How to use

The most straightforward way to use `urlparse4` is to replace your imports of `urlparse` with this:

```
import urlparse4 as urlparse
```

https://github.com/commonsearch/urlparse4

Autres analyses

- Détection de langue : *cld2*
- Détection charset : *cchardet* + metatags/headers
- Cleaning titres & metadata

Index

Quick start

Whoosh is a library of classes and functions for indexing text and then searching the index. It allows you to develop custom search engines for your content. For example, if you were creating blogging software, you could use Whoosh to add a search function to allow users to search blog entries.

A quick introduction

```
>>> from whoosh.index import create_in
>>> from whoosh.fields import *
>>> schema = Schema(title=TEXT(stored=True), path=ID(stored=True), content=TEXT)
>>> ix = create_in("indexdir", schema)
>>> writer = ix.writer()
>>> writer.add_document(title=u"First document", path=u"/a",
...                     content=u"This is the first document we've added!")
>>> writer.add_document(title=u"Second document", path=u"/b",
...                     content=u"The second one is even more interesting!")
>>> writer.commit()
>>> from whoosh.qparser import QueryParser
>>> with ix.searcher() as searcher:
...     query = QueryParser("content", ix.schema).parse("first")
...     results = searcher.search(query)
...     results[0]
...
{"title": u"First document", "path": u"/a"}
```

https://pypi.python.org/pypi/Whoosh/

Search with Apache So @ select provider

Core (Java) | Solr | PyLucene

Ultra-fast Search Library and Server

Apache Lucene and Solr set the standard for search and indexing performance

Welcome to Apache Lucene

The Apache Lucene™ project develops open-source search software, including:

- Lucene Core, our flagship sub-project, provides Java-based indexing and search technology, as well as spellchecking, hit highlighting and advanced analysis/tokenization capabilities.
- Solr™ is a high performance search server built using Lucene Core, with XML/HTTP and JSON/Python/Ruby APIs, hit highlighting, faceted search, caching, replication, and a web admin interface.
- PyLucene is a Python port of the Core project.

DOWNLOAD

Apache Lucene 6.2.1

DOWNLOAD

Apache Solr 6.2.1

http://lucene.apache.org/

```
➜ ~ curl -X POST "http://localhost:39200/confs/pycon/1" -d '{"title": "PyCon France", "country": "FR"}'
{"_index":"confs","_type":"pycon","_id":"1","_version":4,"_shards":{"total":1,"successful":1,"failed":0},"created":false}%
➜ ~
➜ ~ curl -X POST "http://localhost:39200/confs/pycon/2" -d '{"title": "PyCon U.S.", "country": "US"}'
{"_index":"confs","_type":"pycon","_id":"2","_version":3,"_shards":{"total":1,"successful":1,"failed":0},"created":false}%
➜ ~
➜ ~ curl -X GET "http://localhost:39200/confs/pycon/_search?q=france&pretty=1"
{
  "took" : 18,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "failed" : 0
  },
  "hits" : {
    "total" : 1,
    "max_score" : 0.5,
    "hits" : [ {
      "_index" : "confs",
      "_type" : "pycon",
      "_id" : "1",
      "_score" : 0.5,
      "_source" : {
        "title" : "PyCon France",
        "country" : "FR"
      }
    } ]
  }
}
➜ ~
```

https://www.elastic.co

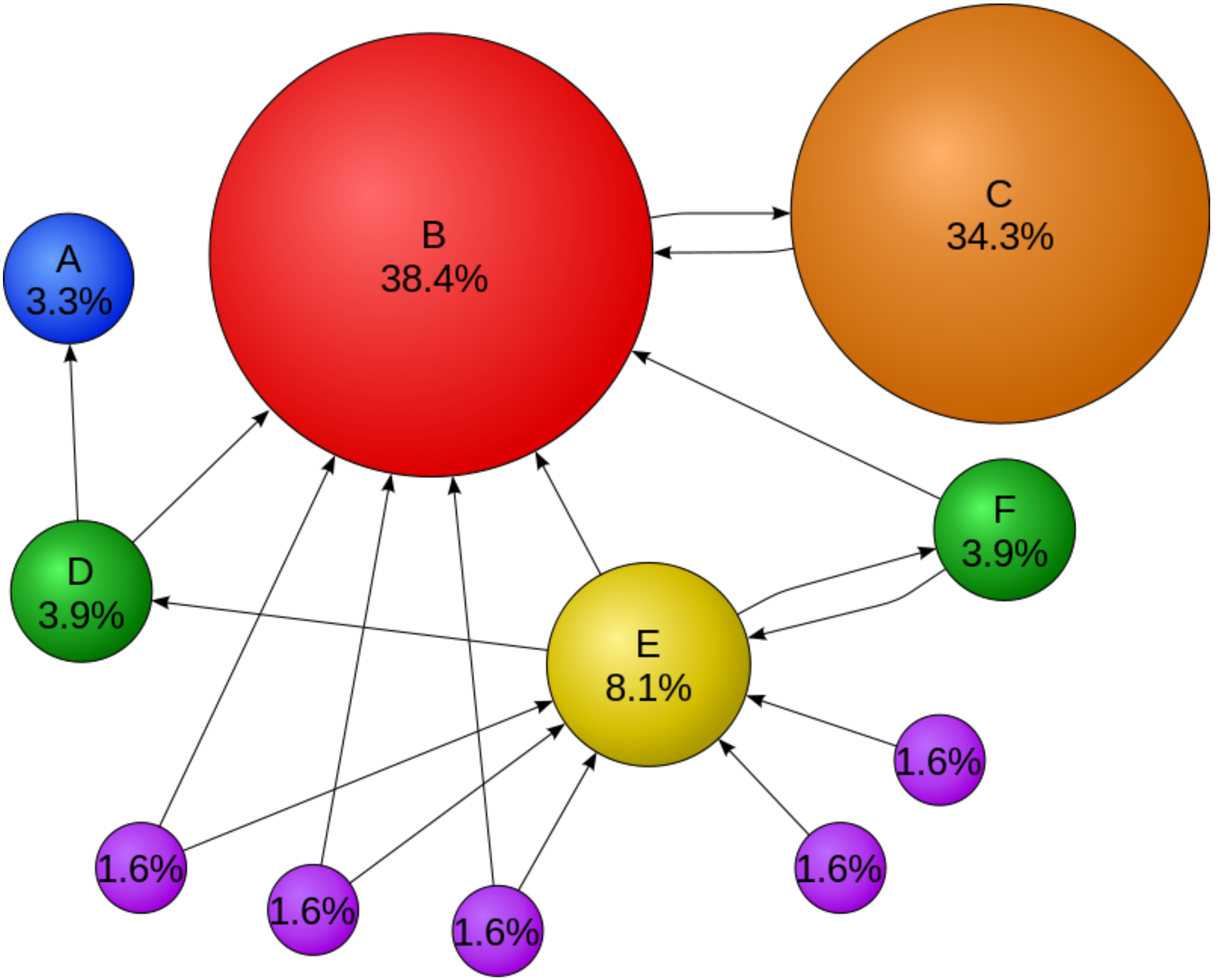
Ranker

Formule du ranking

rank = f(static_score , dynamic_score(query))

Alexa
DMOZ
Blacklists
PageRank
...

ElasticSearch & Lucene
TF-IDF
BM25


A
3.3%
B
38.4%
C
34.3%
D
3.9%
E
8.1%
F
3.9%
1.6%
1.6%
1.6%
1.6%
1.6%

Tutorial: Running PageRank on the Web

This tutorial get you through all the steps required to run PageRank on billions of pages using Common Search's codebase and tools such as Apache Spark and AWS.

1. Prerequisites

You should go through our Analyzing the web with Spark on EC2 first, to install the required software, understand the basic concepts of our pipeline, and run a simpler job first, at least on your local machine.

You should also be familiar with basic Graph theory.

2. Dumping the Web Graph

Before computing PageRank, we need to parse all the link in our corpus and save them as a directed graph.

(In some cases, you can actually skip this step by using one of the dumps we publish directly.)

To dump the web graph, we are doing to use the `webgraph` plugin. Here is how you would dump it for the first 400 URLs from Common Crawl, at the host level:

```
spark-submit --verbose \
    /cosr/back/spark/jobs/pipeline.py \
    --source commoncrawl:limit=4,maxdocs=100 \
    --plugin plugins.webgraph.DomainToDomainParquet:path=out/webgraph/ \
    --stop_delay 600
```

This will actually create 2 subdirectories in `out/webgraph/`: one for the vertices and one for the edges. Both dumps will be stored as Apache Parquet format, so that we can easily reuse them in the next step.

You might notice this command will go over the source documents multiple times. This shouldn't be a big issue with so few

https://about.commonsearch.org/developer/get-started

Searcher

```
# https://www.elastic.co/guide/en/elasticsearch/guide/current/multi-field-search.html
es_query = {
    "must": {
        "multi_match": {
            "query": q,
            "minimum_should_match": "-25%",
            "type": "cross_fields",
            "tie_breaker": 0.5,
            "fields": ["title^3", "body", "url_words^2", "domain_words^4", "paid_domain_words^8"]
        }
    }
}
```

https://github.com/commonsearch/cosr-back/blob/master/cosrlib/searcher.py

Go version:
https://github.com/commonsearch/cosr-front

Frontend

common search: python EN

About 41380 results

Welcome to **Python**.org
www.python.org
The official home of the **Python** Programming Language

Dive Into **Python**
www.diveintopython.net
This book lives at . If you're reading it somewhere else, you may not have the latest version.

The Eric **Python** IDE
eric-ide.python-projects.org
Eric is a full featured **Python** editor and IDE, written in **Python**. It is based on the cross platform Qt gui toolkit, integrating the highly flexible Scintilla...

Starship
www.python.net
The home of pythonistas

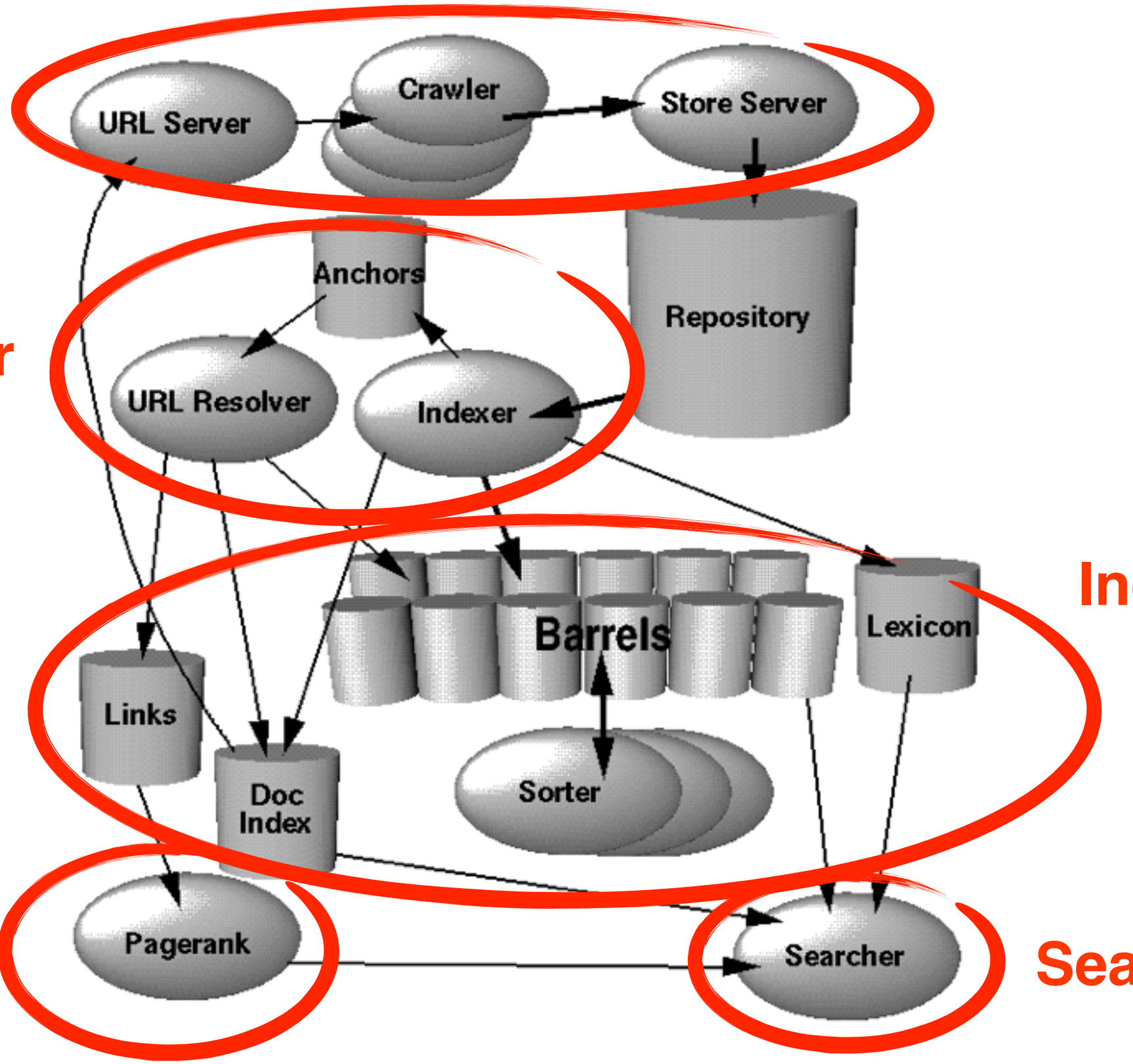
Tutorials, **Python** Courses: Online and On Site
python-course.eu
Free comprehensive online tutorials suitable for self-study and high-quality on site **Python** courses in Europe, Canada (Toronto) and the US

learning **python** | one man's journey into **python**…
www.learningpython.com
one man's journey into **python**...

https://uidemo.commonsearch.org

Crawler

Parser



Index

Ranker

Searcher

The Anatomy of a Large-Scale Hypertextual Web Search Engine (1998)

http://infolab.stanford.edu/~backrub/google.html

Qu'est-ce qui manque ?

Architecture

- 2-pass search (host clustering, result diversity)
- Indexation continue
- Infoboxes
- ~~Pubs~~
- Verticaux (images, vidéos, news, science, ...)
- ...

Encore plus de fun

Spam / Relevance

Sustainability

Outreach

API

...

Ca vous tente?

https://about.commonsearch.org/contributing

https://github.com/commonsearch

contact@commonsearch.org

slack.commonsearch.org