Internal mesh optimization
Semantic linking and siloing
Big data

DUPREY Stefan

Cdiscount

Plan

1. Heuristic
2. Metaheuristics
3. Shrinking our universe with siloing and semantic similarity
4. Same problem with an e-commerce flavor
5. Prototyping example
6. Big data implementation : Neo4j and Spark GraphX
7. Video and paper link

1. Heuristic

Some notations

Let $N \in \mathbb{N}$ be the number of nodes in our mesh.
Let $(X_i)_{i \in \{1,\ldots,N\}}$ be the vertices (URLs) of our oriented graph.
Let $(G_{ij}) \in \{0,1\}^{N \times N}$ be the adjacency matrix of our oriented graph.
Let here define f, a given data per URL, which gives a potentiality metrics for our vertices.

$$\begin{aligned} f \quad : \quad (X_i)_{i \in \{1,\ldots,N\}} &\rightarrow \mathbb{R}^+ \\ x &\mapsto f(x) \end{aligned} \tag{1}$$

$$f(X_i) = PR_{ext}(X_i) + \epsilon(X_i) \tag{2}$$

In-rank

We restrain the universe to our site where we compute the standard page-rank.

Initialization :

$$\forall u \, PR(u) = \frac{1}{N} \tag{3}$$

Iterative computation :

$$PR(u) = \frac{(1-c)}{N} + c \times \sum_{v \to u} \frac{PR(v)}{card(\{v \to u\})} \tag{4}$$

Our heuristic

We want here to optimize the adequacy of our mesh (X_i) to our potentiality vector f. We here postulate the following heuristic to assess the relevance of a mesh :

$$\max_{(G_{ij})\in\{0,1\}^{N\times N}} \left\{ \sum_{i=1}^{N} f\left(X_i\right) \times pageRank(X_i) \right\} \tag{5}$$

2. Metaheuristics

Exhaustive brute force doesn't work

For a $N = 10^6$ millions URLs web site, we have 2^{N^2} with a 2048 bits mantissa, 256 bits exponent

$2^{{10^6}^2} =$

9.5762442314927432848050594956989483747127095675192905698213128517073583274396016675898

714705184143146468453752442806484690561169975498415015777492655947375270159476651418975

300707658547568802353384879419803574730952480197774380552040662758127609571333683703207

910070247048194459504686986124786492353387550318495241621572271925127288273993787778380

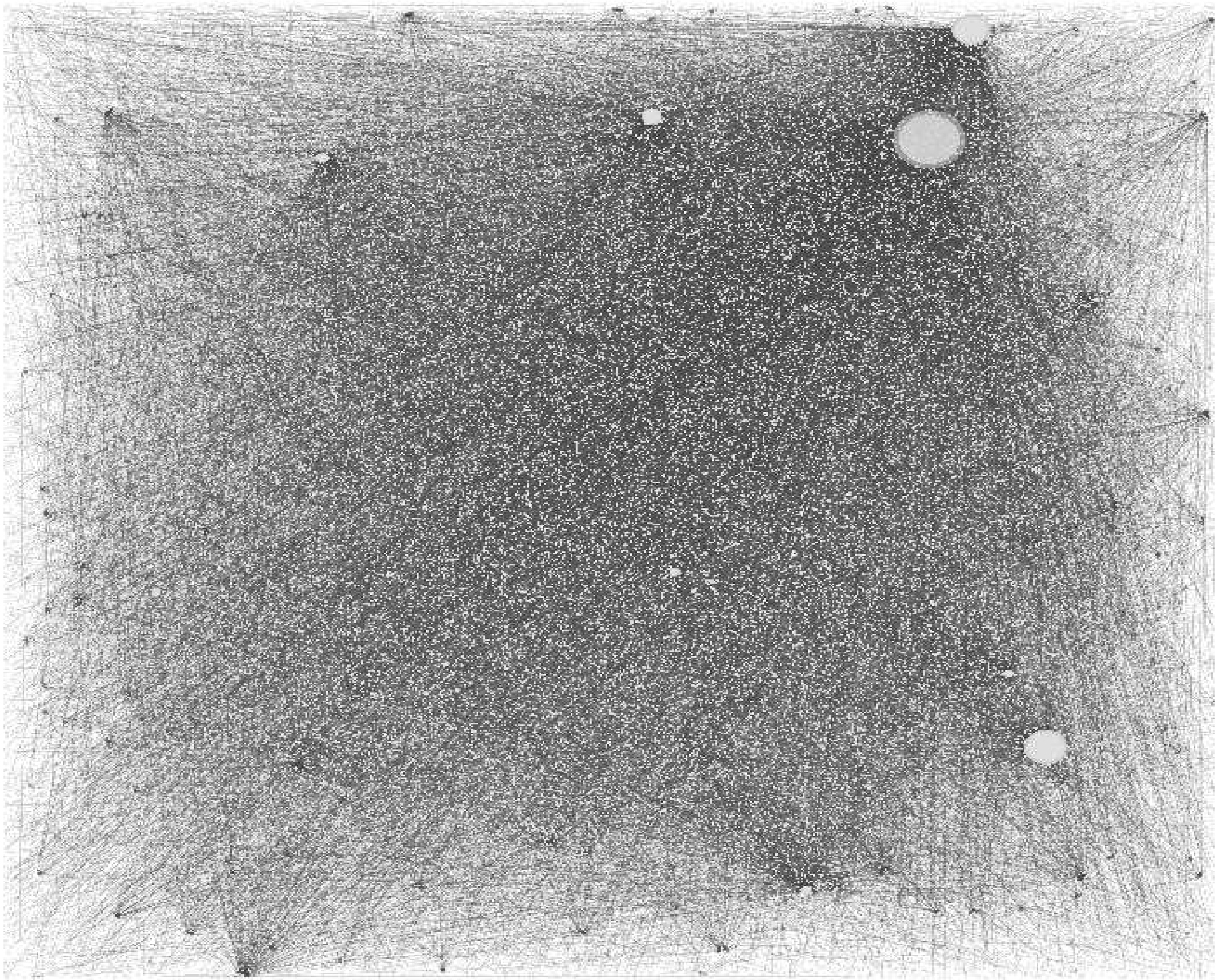
450774809611395810191417363401889038757182279484019203870177413318113073911418463615759

647977538478560166958988721048687854280187283661925937530017243461145905573802314471888

491758757162677684017424597014433418179115289463552630751896559312213624470617453325056

5836008e+301029995663

Picturing our smallest store : jewelery



Metaheuristics

We here have to maximize over all possible graphs. We have to find a clever global optimization methodology : metaheuristics such as global search, multistart, particle swarm, simulated annealing or genetic algorithm are all good candidates.

What is a genetic algorithm

- Genetic algorithm mimics evolutionary biology to find approximate solutions to optimization problems
- Start with an initial generation of candidate solutions that are tested against the objecive function (fitness of the individual)
- Subsequent generations evolve from the first through selection, crossover and mutation
- The individual that best minimizes the given objective is returned as the ideal solution

Why a genetic algorithm

- Lots of local minima to avoid
- Non continuous universe, constraints and objective
- Problem with noise and non-smooth data

Cleverly evolving through our universe

Child spawning from 2 parents crossover

(0 0 0 0 1 0 1 1 1 1 0 1 0)

(1 1 0 1 1 0 1 1 1 1 1 0 0)

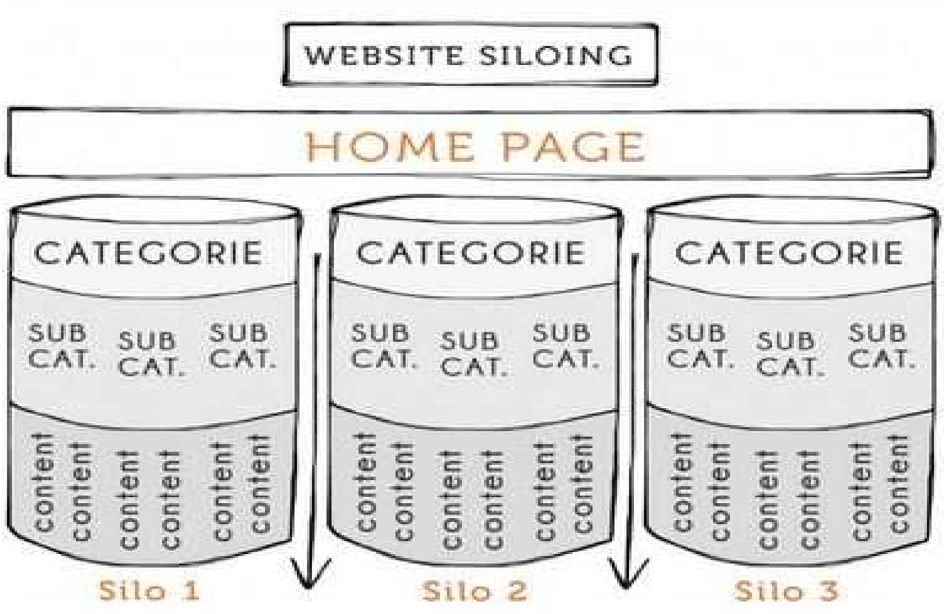
(* * 0 * 1 0 1 1 1 1 * * 0)

Mutation of an individual

(1 1 0 * 1 0 1 1 * 1 1 0 0)

3. Shrinking our universe with siloing and semantic similarity

Siloing and links categorizing



metalist1/metalist2 metalist1/list1

(001010 .. 00110 ..... 10100)

Semantic similarity

We slack our universe by allowing links only between semantically similar URLs :

$$\max_{(G_{ij})\in\{0,1\}^{N\times N}\ G_{ij}=0\ if\ CS(ij)\leq t} \left\{ \sum_{i=1}^{N} f\left(X_i\right) \times PR(X_i) \right\} \tag{6}$$

where $CS(ij)$ is a semantic distance between the two linked pages i and j. $CS(ij)$ can be defined very easily as the scalar product of the tf/idf vectors of the product descriptions whose weight are defined by the well known formula :

$$w_{ik} = \frac{tf_{ik} \log\left(\frac{N}{n_k}\right)}{\sqrt{\sum_{k=1}^{t}(tf_{ik})^2 \log\left(\frac{N}{n_k}\right)^2}} \tag{7}$$

4. Same problem with an e-commerce flavor

$$\sum_{i \in Keywords} SV\left(i\right) \times CTR\left(position\left(i\right)\right) \times PR\left(i\right) \times P\left(i\right) \qquad (8)$$

where $position\left(i\right)$ is the estimated position in search engine results coming from the modification of our new mesh,

$SV\left(i\right)$ is the search volume for the keywords i estimated by the search engine.

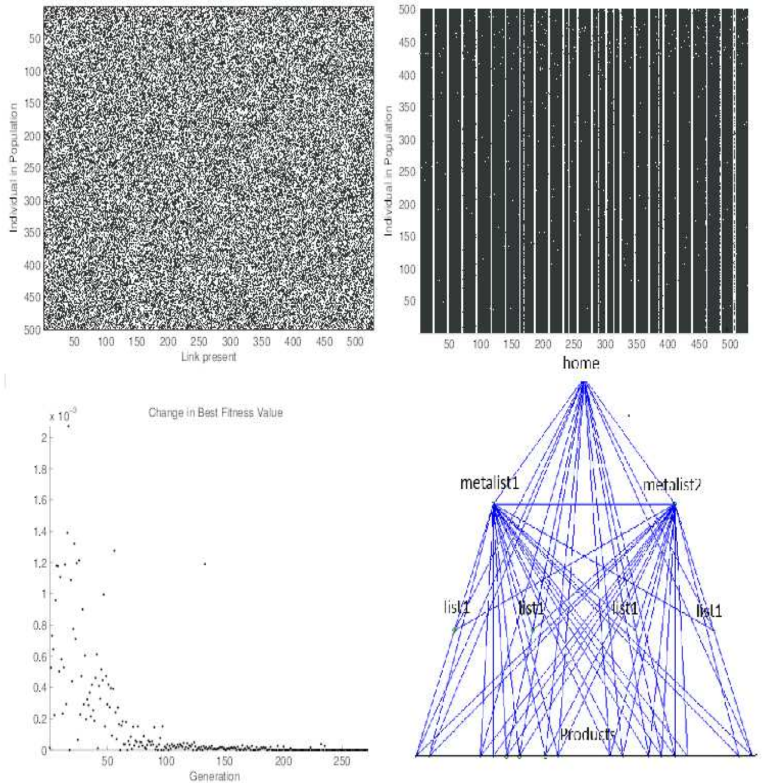
and $CTR\left(i\right)$ is the click through rate for an URL at the position place in the search engine results.

5. Prototyping example

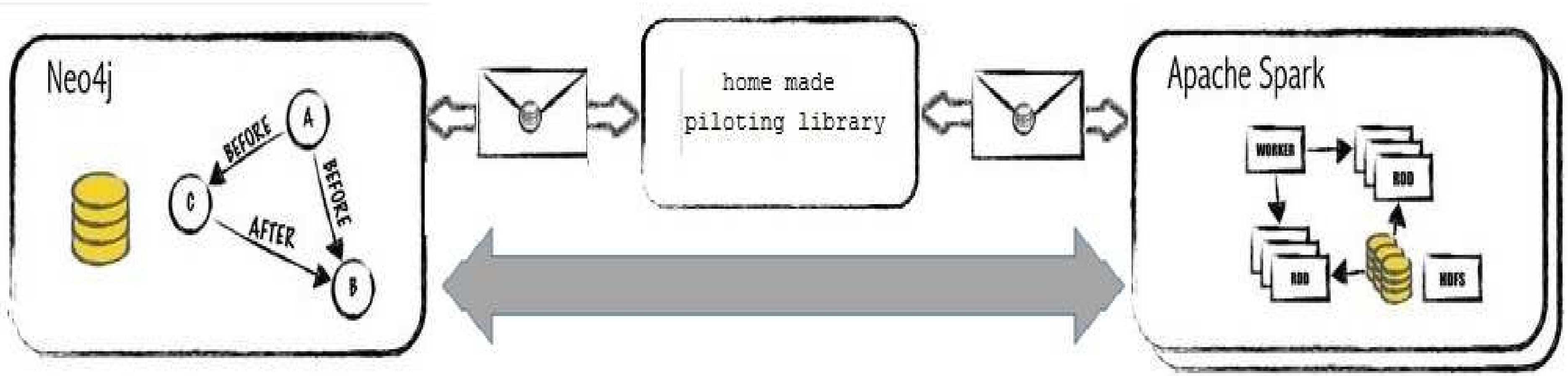
(X_i) ='home','metalist1','metalist2','list1','list2','list3','list4','product1','product2',...,'product32' ;

$f\ (X_i)$=[100, 80, 55, 40, 35, 25, 20, 4, 5,...,3] ;

Results



6. Big data implementation : Neo4j and Spark GraphX



7. Video and paper link

http ://sduprey.github.io/page_rank_optimization_video.html

http ://sduprey.github.io/article_sduprey_iswag_02_06_2015.pdf

http ://sduprey.github.io/PRESENTATION_ISWAG.pdf