The Noise Cluster Model, a Greedy Solution to the Network Community Extraction Problem

Etienne Côme*, Eustache Diemert†

* IFSTTAR - Bâtiment Descartes 2,
2, Rue de la Butte verte,
93166 Noisy le Grand Cedex, France
etienne.come@ifsttar.fr
† BestOfMedia Group,
485 avenue de l'Europe,
F-38330 Montbonnot, France
ediemert@bestofmedia.com

Résumé

Cet article présente un algorithme permettant d'extraire une communauté de nœuds densément connectés dans un graphe. La solution proposée à ce problème s'appuie sur une approche semi-supervisée au sens où un ensemble de graines (nœuds appartenant à la communauté à extraire) doit être fourni. En partant de ces graines l'algorithme explore le graphe et décide d'ajouter ou non les nouveaux nœuds rencontrés à la communauté en utilisant deux tests basés sur une version contrainte du modèle de mélange de graphes de type Erdös-Rényi [7]. Ce modèle simple sera appelé "noise cluster model". Une méthode d'estimation en ligne [23] est utilisée pour mettre à jour les paramètres du modèle tout au long de la procédure d'extraction de la communauté. Cette approche est donc locale au sens où elle présente une complexité dependant principalement de la taille de la communauté à extraire et indépendante de la taille du graphe complet, ce qui permet d'appliquer celle-ci sur des graphes de tailles quelquonques. Finalement, des expériences sur des communautés réelles de blogs seront présentées pour juger de la pertinence de l'approche proposée.

Mots-clés *: clustering de graphe, extraction de communautés, apprentissage semi-supervisé*

Abstract

This paper presents an algorithm designed to extract one community (a collection of vertices that are densely connected amongst themselves) from a graph given some seeds (nodes known to belong to the community). Starting from these seeds nodes, new nodes will be added to the community by selecting them among the successors of the

current community members. The process used to select the community members among the successors is based on a generative model closely related to Erdös-Rényi mixture [7] called the Noise Cluster Model. An on-line estimation procedure [23] is used to update the model parameters during the community extraction process. This approach is local, the complexity is mainly influenced by the community size and does not depend upon the graph size. This method can therefore be used to deal with huge graphs. Eventually, experiments on real blog communities will show the interest of such an approach.

***Key-words**: graph clustering, community extraction, semi-supervised, noise cluster model*

1 INTRODUCTION

A community could be loosely described as a collection of vertices within a graph which are densely connected amongst themselves while being loosely connected to the rest of the graph. Community detection in complex networks has attracted a lot of attention in recent years (for a review, see [9]). In fact, detecting communities or modules can be a way to identify interesting substructures which could correspond to those parts of the networks which have specific properties. Communities may correspond for example to specific functions in biological networks [12] or to specific topics in web pages networks [8, 1] (the experimental part of the paper will also highlight this fact). Therefore, identifying communities may help in understanding more deeply the structure of the analysed networks. As in biology, where it is widely believed that biological networks of genes or proteins present a modular structure which is the results from evolutionary constraints and plays a crucial role in biological functions [20, 19, 14]. Other relevant examples of interesting community structures can be found in social networks [11, 16] or food webs [15].

The main approaches to identify communities in networks are based on graph clustering algorithms which take as input a whole graph and supply a partition of the vertices which optimize an objective function such as the widely used Newman *modularity* [17]. Similar solutions more linked to the proposal of this paper use the Erdös-Rényi mixture model [7, 23, 24], also called *block models* [13, 21] to derive the objective function. All these methods search for all the communities of the network and are therefore global, their complexities scale with the size of the graph.

The problem addressed here is slightly different : the aim of the proposed algorithm is to extract only one community of interest from the network, the others communities being considered as useless. To do so the algorithm will be supplied with seeds nodes defined as nodes known to belong to the community of interest. Such an algorithm can be useful for example to extract a set of web pages on a topic of interest using few seeds pages. Taking into account the local nature of the problem (the structure of the networks

outside the target community being of no interest to solve the problem), this paper proposes a local algorithm built over the Erdös-Rényi mixture model to extract the community which encloses the seeds. This algorithm has a complexity (in term of memory space and computational time) which is mainly influenced by the size of the extracted community and do not depend upon the graph size.This property is of interest and enables the use of such a solution on very big graphs such as the World Wide Web graph. Experiments will highlight this fact by using this algorithm to extract blog communities dealing with specific topics.

Other local procedures have already been proposed to extract one community starting from seeds nodes. Bagrow & al [3] propose a technique which relies upon growing a breadth-first tree outward from one seed node, until the rate of expansion (proportion of edge found at the current level which lead to nodes which are yet unknown) falls below an arbitrary threshold. This simple solution is interesting. However, since all the nodes found at one level of the breadth-first tree are added to the community (if the rate of expansion is below the threshold), it will succeed in extracting the community only if the source vertex is equidistant from all parts of its enclosing community boundary. The seed must therefore be carefully chosen or multiple seeds used and the results combined (this second solution is advocated by the authors). Another solution proposed in [6] is based on the greedy optimization of a quantity called *local modularity*. This quantity involves a specific set of nodes called *boundary*. This set is defined as the set of nodes that have at least one neighbor in the set of yet unknown nodes. Local modularity is then defined as the number of edges between this set and the set of known nodes over the total number of edges with one extremity in this set. The greedy optimization of this quantity simply adds the unknown node which gives the largest increase (or the smallest decrease) of the local modularity to the community until a predefined number of nodes is reached. As with the previous solution, only one node is used as seed, which is different from our solution. Furthermore, here the optimized criterion is derived from an *ad hoc* definition and no solution to automatically stop the extraction process is supplied (the number of nodes to extract must be supplied by the user). Other solutions to the community extraction problem use conductance and random walks [2] or combinatorial algorithms [22] to define the extraction procedure, however these solutions present complexities that scale linearly with the size of the graph, whereas our solution scale with the size of the community to extract.

The road map of the paper is the following, first some background on Erdös-Rényi mixture model will be supplied in section 2. Then, the constrained version of this model used in the paper will be detailed in section 3. Eventually, section 4 presents the proposed local algorithm, and section 5 details preliminary experiments on real blog community extraction problems.

2 BACKGROUND ON ERDÖS-RÉNYI MIXTURE MODEL

Formally, the graph clustering problem is set-up in the Erdös-Rényi mixture model with the help of two sets of random variables with the following meaning (capital letters denote random variables whereas non capital letters denote realizations of the same random variables) :

– $X_{ij} \in \{0, 1\}$ are binary variables indicating the presence or the absence of an edge from i to j :

$$x_{ij} = \begin{cases} 1, \text{ if there is a link from } i \text{ to } j \\ 0, \text{ otherwise.} \end{cases} \tag{1}$$

– $Z_j \in \{1, \ldots, K\}$ are latent variables encoding cluster membership of vertex j among the K possible clusters, such that :

$$z_j = k, \text{ if } j \text{ belongs to cluster } k. \tag{2}$$

Directed graphs will be considered in this paper. Therefore we will consider that x_{ij} may differ from x_{ji}. These variables have the following distributions in this model :

$$Z_j \overset{i.i.d}{\sim} \mathcal{M}(1, \gamma), \quad \forall j \in \{1, \ldots, N\} \tag{3}$$

$$X_{ij}|Z_i = k, Z_j = l \overset{i.i.d}{\sim} \mathcal{B}(\pi_{kl}), \quad \forall i, j \in \{1, \ldots, N\}, \tag{4}$$

where $\mathcal{M}$ denotes the Multinomial distribution and $\mathcal{B}$ the Bernoulli distribution. This generative model has therefore the following interpretation :

1. draw the cluster of each node according to the probabilities γ
2. add an edge between i and j with a probability π_{kl} if i belongs to cluster k and j belongs to cluster l.

Therefore, when $\pi_{kk} \gg \pi_{kl}, \forall k \neq l$, clusters correspond to dense components in the graph, and this model can be used to recover the community structure of a graph [23, 21]. We propose to use in the context of community extraction a simpler model with less parameters. We present shortly this model in the next section. This constrained model is dedicated to the case where the graph contains only one community and background noise with no specific structure.

3 THE NOISE CLUSTER MODEL

We will consider only two mixture components, one for the community which encloses the seeds and one for the nodes that do not belong to that community, that will be called the *noise component*. We will therefore use

only one Bernoulli variable Z_i to deal with cluster membership of vertex i, which is defined as :

$$z_i = \begin{cases} 1, \text{ if } i \text{ belongs to the community of interest} \\ 0, \text{ if } i \text{ belongs to the noise component} \end{cases} \tag{5}$$

The model is a constrained version of the block model and takes the following simple form :

$$Z_i \overset{i.i.d}{\sim} \mathcal{B}(\gamma), \quad \forall i \in \{1, \ldots, N\} \tag{6}$$

$$X_{ij}|Z_i \times Z_j = 1 \overset{i.i.d}{\sim} \mathcal{B}(\alpha), \quad \forall i, j \in \{1, \ldots, N\} \tag{7}$$

$$X_{ij}|Z_i \times Z_j = 0 \overset{i.i.d}{\sim} \mathcal{B}(\beta), \quad \forall i, j \in \{1, \ldots, N\} \tag{8}$$

We therefore have only three parameters $\theta = (\alpha, \beta, \gamma)$, γ is the prior probability of the community, α is the probability that two nodes from the community are linked and β is the probability that tunes the noise cluster behaviour. This simple model is sufficient to represent the community structure that we are interested in, provided that $\alpha \gg \beta$. Let us introduce some notations and properties of this model which will be used in the sequel.

Definition 1
Let d_j be node j degree with community members, d_j^{in} node j in-degree with community members and d_j^{out} node j out-degree with community members :

$$d_j^{in} = \sum_{i:z_i=1} x_{ij}, \quad d_j^{out} = \sum_{i:z_i=1} x_{ji}, \quad d_j = \sum_{i:z_i=1} (x_{ij} + x_{ji})$$

Definition 2
Let p_{N+1}^i be the community membership posterior probabilities of a new node given only its in-links and the cluster membership of the first N nodes :

$$p_{N+1}^i = \mathbb{P}(Z_{N+1} = 1 | x_{i(N+1)}, z_i, \forall i \in \{1, \ldots, N\}).$$

Let p_j^{io} be the community membership posterior probabilities of a new node given its in-links and out-links and the cluster membership of the first N nodes :

$$p_{N+1}^{io} = \mathbb{P}(Z_{N+1} = 1 | x_{i(N+1)}, x_{(N+1)i}, z_i, \forall i \in \{1, \ldots, N\}).$$

Proposition 3.1
Community membership posterior probabilities p_j^i, and p_j^{io} depend only on parameters α, β, γ and d_j^{in}, d_j respectively and are given by :

$$p_{N+1}^i = \frac{\gamma \alpha^{d_{N+1}^{in}} (1-\alpha)^{(N_c - d_{N+1}^{in})}}{\gamma \alpha^{d_{N+1}^{in}} (1-\alpha)^{(N_c - d_{N+1}^{in})} + (1-\gamma) \beta^{d_{N+1}^{in}} (1-\beta)^{(N_c - d_{N+1}^{in})}}, \tag{9}$$

$$p_{N+1}^{io} = \frac{\gamma\alpha^{d_{N+1}}(1-\alpha)^{(2N_c-d_{N+1})}}{\gamma\alpha^{d_{N+1}}(1-\alpha)^{(2N_c-d_{N+1})} + (1-\gamma)\beta^{d_{N+1}}(1-\beta)^{(2N_c-d_{N+1})}}, \quad (10)$$

with $N_c = \sum_{i=1}^{N} z_i$ the community size. See appendix 6 for the derivation of these equations.

The probabilities p_{N+1}^{i} and p_{N+1}^{io} depend only on graph structure through d_j^{in} (the number of in-goings links from the community members) and d_j (the total of links with community members) respectively. The number of links shared with the noise component is irrelevant. This property comes from the constraints imposed by the noise cluster model over the general Erdös-Rényi mixture model.

Figure 1 gives an example of this conditional law. As expected, this quantity increases with d_j^{in} (with $\alpha \gg \beta$). More links from the community give therefore a higher probability of belonging to the community.

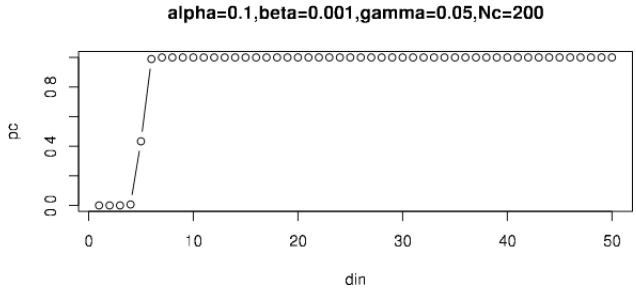

FIG. 1 – Values of p_{N+1}^{in} with respect to d_{N+1}^{in} with α=0.1, β=0.001,γ=0.05 and N_c=200.

Starting from this simple model, we will describe an online, greedy algorithm that adds new nodes to the community from the community successors.

4 LOCAL ALGORITHM DESCRIPTION

As explained in the introduction, the algorithm is supplied with seeds nodes. These seeds are considered to belong to the community with certainty, and along its path the algorithm add new nodes to the community by looking at the current community members out-going links.

The algorithm proceeds one vertex at a time in a breadth first fashion, but uses the previous generative model to decide which found node to add to the community. A first test, which uses only in-links information, is performed to find new nodes that may belong to the community. Such nodes are then added to the queue of nodes which require further investigation. When a node

succeeds in this first test, another test (which takes into account the in-going and out-going links of the node) is performed to decide whether to add it permanently to the community. This process is repeated until no more nodes are accepted by the first test. Throughout the community extraction process the three model parameters are updated using an on-line estimation strategy [23]. The core of the algorithm is the two tests used to decide to add or not one node to the community or not and the on-line parameters estimation procedure. The two tests are derived directly from equations (9) and (10). We describe them shortly and give some insights into the on-line estimation procedure.

4.1 Community Membership tests

When only in-links are known it is natural to decide that j belongs to the community when $p^i_{N+1} > s$. The threshold s can be defined by default to 0.5 but it can also be interesting to use more strict values such as 0.8 or 0.9 when one wants to take less risk of contaminating the extracted community with noise. Starting form equation (9) we may rewrite the test in terms of d^{in}_{N+1} (see appendix 6 for the details) :

$$d^{in}_{N+1} > d^{in}_{min}, \tag{11}$$

with d^{in}_{min} equals to :

$$d^{in}_{min} = \left\lfloor \frac{\log\left(s(1-\beta)^{N_c}(1-\gamma)\right) - \log\left((1-s)(1-\alpha)^{N_c}\gamma\right)}{\log\left(\alpha(1-\beta)\right) - \log\left((1-\alpha)\beta\right)} \right\rfloor$$

Figure 2 presents the evolution of d^{in}_{min} with respect to the community size N_c which has a simple step profile. Similar expressions can be obtained for the test which uses in and out links using equation (10) which is performed in a second step when the node out-links have been retrieved.

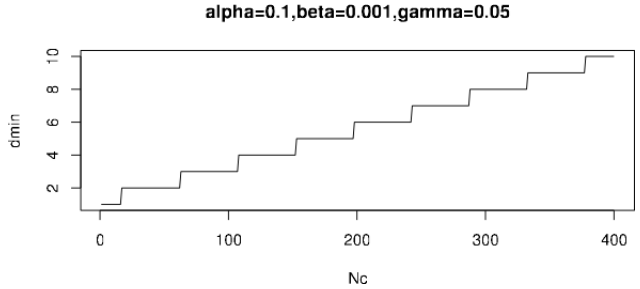

FIG. 2 – d_{min} evolution with respect to the community size N_c with $\alpha = 0.1$, $\beta = 0.001, \gamma = 0.05$ and $s = 0.5$.

4.2 Parameters estimation

This section describes how the incremental classification version of the EM (CEM) algorithm, proposed by [23], can be adapted to estimate the previous model parameters during the community extraction process. The proposed algorithm differs from [23], by the order in which nodes are processed and by the used stopping criterion. In the classical online CEM algorithm the whole graph is processed and node are randomly ordered, whereas in our proposal only a small portion of the graph is processed and nodes are processed according to their distances to the seeds. We first present the criterion used to estimate the parameters, known as *classification likelihood*, then the estimation procedure itself. In the case of a full adjacency matrix, the classification log-likelihood is defined as :

$$\begin{aligned} L_c(\theta, \mathbf{X}, \mathbf{Z}) &= \sum_i z_i \log(\gamma) + \sum_i (1 - z_i) \log(1 - \gamma) \\ &+ \sum_{i,j:i\neq j} z_i \times z_j \times x_{ij} \log(\alpha) + \sum_{i,j:i\neq j} z_i \times z_j (1 - x_{ij}) \log(1 - \alpha) \\ &+ \sum_{i,j:i\neq j} (1 - z_i \times z_j) \times x_{ij} \log(\beta) + \sum_{i,j:i\neq j} (1 - z_i \times z_j) \times (1 - x_{ij}) \log(1 - \beta) \end{aligned}$$

with $\mathbf{Z} = \{z_1, \ldots, z_N\}$, $\mathbf{X} = \{x_{ij} : i \neq j; i, j \in \{1, \ldots, N\}\}$, and $\theta = (\gamma, \alpha, \beta)$ the parameters vector.

If the partition $\mathbf{Z} = \{z_1, \ldots, z_N\}$ is known and with a square adjacency matrix of size $N \times N$, the parameter vector maximizing the classification likelihood is found by setting to zero the derivative of the classification log-likelihood with respect to each parameters. They are therefore given by :

$$\hat{\gamma} = \frac{N_c}{N}, \tag{12}$$

$$\hat{\alpha} = \frac{1}{N_c^2} \sum_{i,j=1,\, i\neq j}^{N} (z_i \times z_j) x_{ij}, \tag{13}$$

$$\hat{\beta} = \frac{1}{N_{\bar{c}} \times (N + N_c)} \sum_{i,j=1,\, i\neq j}^{N} (1 - z_i \times z_j) x_{ij}, \tag{14}$$

with $N_{\bar{c}}$ the number of nodes that do not belong to the community, $N_{\bar{c}} = \sum_{i=1}^{N} (1 - z_i)$ and N the total number of nodes.

However, the partition $\mathbf{Z} = \{z_1, \ldots, z_N\}$ is unknown and must also be estimated, an on-line alternating optimization solution can be used to solve this problem. For this purpose the two previous tests are used to estimate the partition for every new nodes and equations (12, 13, 14) are used to update the parameters after each test. Such solution is sub-optimal since N_{barc} will be underestimated but works well in practice and is really fast. Eventually, it is important to note that equations (12, 13 and 14) can be computed incrementally to avoid unnecessary calculus, see [23] for details.

4.3 Local greedy algorithm for community extraction

All the pieces put together lead to the local greedy algorithm that we propose for community extraction. Algorithm 1 summarise the main steps of this algorithm. We present in the next section some results on several blog communities extraction tasks.

5 EXPERIMENTS : BLOG COMMUNITIES EXTRACTION

An experimental version of the algorithm was developed to deal with networks of HTML documents and used to extract blog communities. This experimental tool is basically a multi-threaded web crawler (which implements the $retrieveoutlinks$ function required by the algorithm) coupled with the community extraction procedure described above. The seeds URLs supplied to the algorithm were taken from a blog portal called Wikio [3] which offers several rankings of blogs for several topics. Theses ranking were used to provide 50 seeds to the algorithm for 4 test communities. All the other inputs of the algorithm were set to default values for all the experiments ($s = 0.5, \alpha^0 = 0.05, \beta^0 = 0.001, \gamma^0 = 0.05$). The goal of the experiments is two-fold :

- **first** we aim to validate that the returned community fulfill the structural definition of a community,
- **second** we want to check that the extracted communities meet the already observed phenomenon [8, 1], *i.e.* that communities of web pages (in the graph sense of densely connected set of vertex) correspond to pages dealing with the same topic.

The communities returned by the algorithm will therefore be analysed first with respect to their structure and in a second step with respect to their content. The estimated model parameters (reported in Table 1) will be used to check that the extracted communities correspond to the loose definition of a community (a collection of vertices within a graph that are densely connected amongst themselves while being loosely connected to the rest of the graph).

- $\hat{\alpha}$ as defined by equation 13 corresponds to the community links density and can therefore be used to validate that vertices of the communities are densely connected.
- $\hat{\beta}$ corresponds to the density of links between community members and non-community members (see equation 14) and can therefore be used to check that community members are loosely connected to the rest of the graph.

Table 1 presents the model parameters estimated by the algorithm and the size of the retrieved communities. This table highlights the fact that the extracted communities have a high internal links density $\hat{\alpha}$ (around 0.02 for all

[3] http ://www.wikio.com, http ://www.wikio.fr

Algorithm 1 Local greedy community extraction based on the noise cluster model

Require: a function to retrieves nodes children : $retrieveoutlinks(node)$
Require: a set of seeds nodes : $seeds$
Require: community membership tests threshold : $s \in [0, 1]$, (default 0.5)
Require: initial value for parameters : $\alpha^{(0)}, \beta^{(0)}, \gamma^{(0)}$

```
{Initialisation}
α ← α^(0), β ← β^(0), γ ← γ^(0)
queue ← seeds
community ← seeds
update : d_min, d^in_min (eq. 11)
{Main loop}
while isnotempty(queue) do
    {Retrieve a community successor}
    node ← dequeue(queue)
    {Retrieve node children}
    outlinks ← retrieveoutlinks(node)
    d^out_node = size(outlinks)
    d_node = d^in_node + d^out_node
    {Test for community membership}
    if d_node > d_min then
        community ← {community, node}
        Nc ← Nc + 1
        {Update children in-links from the community counter}
        for all outlinks do
            target ← target(outlinks)
            d^in_target ← d^in_target + 1
            {Test for community membership using only in-links}
            if d^in_target > d^in_min then
                enqueue(target)
            end if
        end for
    end if
    {Parameters update}
    update : α, β, γ (Eq. 12, 13, 14)
    update : d_min, d^in_min (eq. 11)
end while
return community
```

the communities) and a low $\hat{\beta}$ (around 0.001). The algorithm has therefore succeeded in retrieving a set of densely connected nodes which enclose the seeds and which has furthermore very few links with the rest of the graph. The diameter (the longest shortest path, denoted by *dia*) of the community and the average shortest path length between community members (denoted by *apl* and also reported in Table 1) supply also clues on the strong connections between the communities members : small diameters (between 6 and 8) and small path length (between 2.7 and 3.1).

The communities sizes are reasonable around $1\,000$. Starting from 50 seeds the algorithm was therefore able to expand the community size by a factor between 12 and 36 for the different communities. The sizes of the subnetworks processed by our algorithm N are big, the extracted communities as expected with networks of HTML pages, are therefore small modules with respect to the network size.

	Illustration (Fr)	Scrapbooking (Fr)	Cooking (Fr)	Politics (U.S.A.)
$\hat{\alpha}$	0.01829	0.02955	0.03846	0.02004
$\hat{\beta}$	0.00094	0.00232	0.00209	0.00068
N_c	1 360	701	622	1 808
N	37 101	13 467	16 364	84 702
dia	8	8	6	7
apl	3.059	2.749	2.71	3.014

TAB. 1 – Estimated model parameters $\hat{\alpha}, \hat{\beta}$ and structural statistics for the 4 communities extracted : N_c is the community size, N the total number of vertices seen during the extraction process, *dia* stands for community diameter and *apl* corresponds to the average path length between all the community members.

The second goal of the experiments was to validate the fact that the extracted communities correspond to blogs which deal with the specific topics of the supplied seed. Several, investigations were performed by analysing the blogs contents to confirm this point, and the results for each community are presented in the next subsections.

5.1 Illustration (Fr), community analysis

The first investigation performed on the extracted community, deals with the evaluation of the precision of the algorithm with respect to the seed topic. To evaluate this precision, 100 blogs of the community were manually visited by the authors, and the number of blogs with the same main topic as the seeds (here illustration) was recorded. This estimation of the precision gives 99% for the Illustration community, (only one blogs over the 100 visited blogs was dealing with another topic). We therefore may conclude

that the returned blogs are coherent in terms of topic. However, the language of the analysed blogs was not found to be always French (the seeds language), an important part of the retrieved blogs were written in english. The strong relations between French and American or English illustration community may be explained by the number of French people working in the American animations studios like Disney and Pixar. This point was also checked by analysing the text content of the retrieved communities. To this end, word frequencies in documents (fraction of blogs where the word appears at least once in the front page, denoted by wdf) were computed for each word. Then the Kullback-Leibler divergence between this word document frequency and the document frequency of the same word in a negative class of random blogs was computed. By sorting the words according to their divergence and keeping the best ones, the core vocabulary of each community was extracted. Figures 3 presents word clouds of this core vocabulary for the illustration community. It appears that the words are in adequacy with the community topic, with terms like illustration ($wdf = 25\%$), animation ($wdf = 34\%$), drawing ($wdf = 28\%$), gobelins (a well known art university in France $wdf = 4.6\%$) and so on.

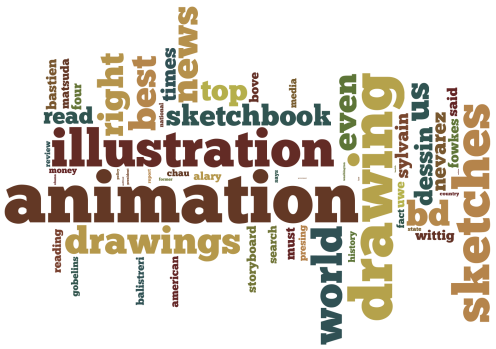

FIG. 3 – Word Clouds for the Illustration (Fr) community. The first 50 words in descending order of their Kullback-Leibler divergence are extracted (between word document frequency in the community and in a negative class of 2000 random blogs, texts have been first preprocessed using stop lists and stemming). Word sizes are proportional to the word document frequencies in the community.

5.2 Scrapbooking (Fr), community analysis

As previously, 100 blogs of the community were manually visited to estimate the precision. This procedure gives a score of 98% for the Scrap-

booking community. Furthermore, for this community, the language of the analysed blogs was found to be always French (the seeds language). Text content of the community blogs was also analysed with the same process as before and the results are displayed in Figure 4. The extracted vocabulary corresponds to the seed topic with for example : scrap ($wdf = 84\%$), scrapbooking ($wdf = 58\%$), tampons ($wdf = 48\%$), ... Furthermore, more than 72% of the blogs urls contain the word scrap. This scores confirm that the returned blogs have for main topic scrapbooking.

FIG. 4 – Word Clouds for the Scrapbooking (Fr) community (same methodology as in Figure 3).

5.3 Cooking (Fr), community analysis

The precision manually estimated over 100 blogs gives 100%, all the visited blogs have therefore for main topic cooking. The community word cloud depicted in Figure 5, highlight the coherence of the retrieved community. The extracted words : cuisine ($wdf = 83\%$ cooking in English), recette ($wdf = 79\%$, recipe in English), chocolat ($wdf = 68\%$, chocolate in English), saveur ($wdf = 44\%$, flavor in English), ... are all connected to the semantic fields of cooking.

5.4 Politics (U.S.), community analysis

The precision estimated for this community is a little less important than for the three first, with 96% of blogs mainly dealing with politics. The vocabulary extracted by the same methodology as before is relevant (see Figure 6) with terms like : senate ($wdf = 29\%$), conservatives ($wdf = 21\%$), democrat ($wdf = 18\%$), pundit ($wdf = 20\%$), terrorism ($wdf = 19\%$), senator ($wdf = 19\%$), medicare ($wdf = 18\%$).

FIG. 5 – Word Clouds for the Cooking (Fr) community (same methodology as in Figure 3).

FIG. 6 – Word Clouds for the Cooking (Fr) community (same methodology as in Figure 3).

Since this community is more complex than the three other, and because of its interesting internal structure, we provide an illustration showing the community network in Figure 7. Different sub-communities can be seen on this figure. Therefore in this case a hierarchy of communities exists, and the presence of sub-communities can be explained by the different political tendencies structuring the political debate in the U.S.A. The algorithm has not extracted only one such community, because the chosen seeds came from different sub-communities.

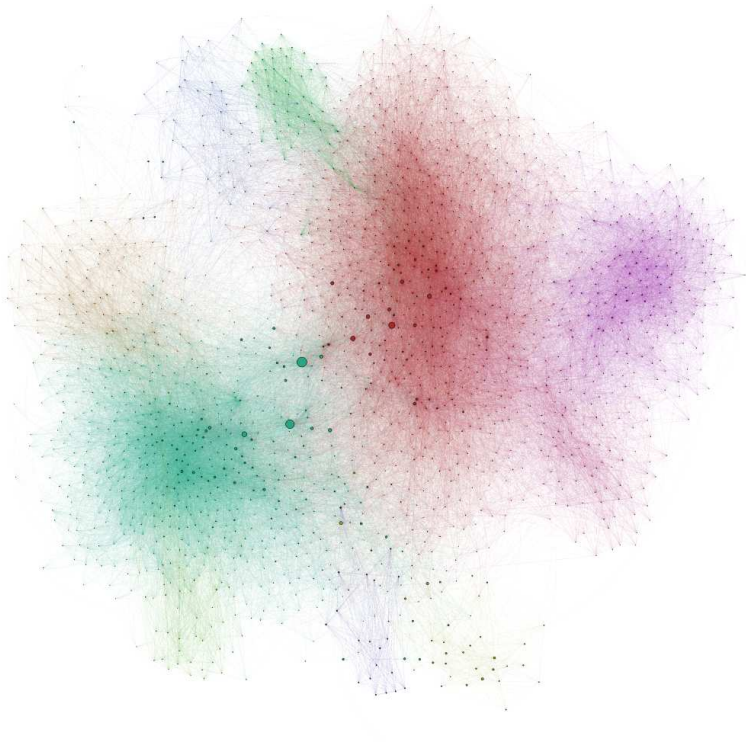

FIG. 7 – Politics (U.S.A.) network drawn using the Fruchterman-Reingold algorithm [10] of the Gephi software [4]. Nodes colours correspond to a modularity clustering of the community, node sizes are proportional to the nodes page-rank [18].

6 CONCLUSION AND FUTURE WORKS

The experimental solution to the community extraction problem proposed in this paper seems relevant. It is quite important to note that a simple, greedy approach is able to extract communities with high precision. Such simplicity and scalability is of great importance when dealing with multi-billion nodes graphs, as is the case with some real world examples like web or online social graphs.

From an experimental point of view, blog community extraction was performed using such a tool with success. However, more work is needed to better understand and evaluate the model.

First, we could find other application domains were different community structures exist with different characteristics [9]. Applying the method to biological systems (*e.g.* protein interaction networks) or online social net-

works and the like may provide clues about the robustness of the approach with respect to the different graphs structures one may find in such different contexts. We could also try to find a generic method to set the initial value of the parameters given these various application domains. Experimenting with structures which are do different may lead to generalize the algorithm in order to make it able to decide if there is only one or several community structure(s) in the explored network. The only drawback of such an approach is the need to have annotated corpora with ground-truth communities.

Second, robustness of the methods to perturbations of the seeds set must be investigated. Comparing the communities extracted by the methods starting from different random samples may help to evaluated this point.

Eventually, we could make use of the related field of graph generation algorithms. The purpose of these algorithms is to be able to generate realistic graphs with predefined output parameters, *e.g.* radius or clustering coefficient. A quite comprehensive overview of this field may be found in [5]. In our case this kind of algorithm may be used to produce synthetic datasets for which we have by construction the ground truth communities. This may greatly help to experiment with our detection algorithm with a broad range of graph structures (by changing the generator algorithm) and variations (by changing the output parameters values).

RÉFÉRENCES

[1] L. Adamic et N. Glance. The political blogosphere and the 2004 us election. In *WWW Workshop on the Weblogging Ecosystem*, 2005.

[2] R. Andersen et K. Lang. Communities from seed sets. In *Proceedings of the 15th International Conference on World Wide Web*, pages 223–232. ACM Press, 2006.

[3] J.P. Bagrow et E.M. Bollt. A local method for detecting communities. *Physical Review E, Statistical, Nonlinear and Soft Matter Physics*, 72(4) :046108, 2005.

[4] M. Bastian, S. Heymann et M. Jacomy. Gephi : an open source software for exploring and manipulating networks. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, 2009.

[5] D. Chakrabarti et C. Faloutsos. Graph mining : Laws, generators, and algorithms. *ACM Computing Surveys*, 38(1), 2006.

[6] A. Clauset. Finding local community structure in networks. *Physical Review E, Statistical, Nonlinear and Soft Matter Physics*, 72(2) :026132, 2005.

[7] J. Daudin, F. Picard et Robin S. A mixture model for random graph. *Statistics and computing*, 18 :1–36, 2008.

[8] G. Flake, S. Lawrence, C. Lee Giles et F. Coetzee. Self-organization and identification of web communities. *Computer*, 35(3) :66–71, March 2002.

[9] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5) :75–174, 2010.

[10] T. M. J. Fruchterman et E. M. Reingold. Graph drawing by force-directed placement. *Software - Practice & Experience*, 21(11) :1129–1164, 1991.

[11] M. Girvan et M. Newman. Community structure in social and biological networks. 2002.

[12] L. H. Hartwell, J. J. Hopfield, S. Leibler et A. W. Murray. From molecular to modular cell biology. *Nature*, 402 :C47–C52, 1999.

[13] J.W. Holland, K.B. Laskey et S. Leinhard. Stochastic block models : First steps. *Social Networks*, 5 :109–137, 1983.

[14] P. Holme, M. Huss et H. Jeong. Subnetwork hierarchies of biochemical pathways. *Bioinformatics*, 19 :532–538, 2003.

[15] A.E. Krause, K.A. Frank, D.M. Mason, Ulanowicz R.E. et Taylor W.W.

[16] D. Lusseau et M.E.J. Newman. Identifying the role that animals play in their social networks. *Proceedings of the Royal Society of London B*, 2004.

[17] M. Newman, et E. Leicht. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences of United States of America*, 104 :9564–9569, 2007.

[18] L. Page, S. Brin, R. Motwani et T. Winograd. The pagerank citation ranking : Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[19] J.A. Papin, J.L. Reed et B.O. Palsson. Hierarchical thinking in network biology : the unbiased modularization of biochemical networks. *Trends Biochemical Sciences*, 29 :641–647, 2004.

[20] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai et Barabasi A.-L. Hierarchical organization of modularity in metabolic networks. *Science*, 297 :1551–1555, 2002.

[21] T. Snijders et K. Nowicki. Estimation and prediction for stochastic block-structures for graphs with latent block structure. *Journal of Classification*, 14 :75–100, 1997.

[22] M. Sozio et A. Gionis. The community-search problem and how to plan a successful cocktail party. In *Proceedings of the 16th ACM SIGKDD Conference On Knowledge Discovery and Data Mining (KDD)*, pages 939–948, 2010.

[23] H. Zanghi, C. Ambroise et V. Miele. Fast online graph clustering via erdos-renyi mixture. *Pattern Recognition*, 41(12) :3592–3599, December 2008.

[24] H. Zanghi, F. Picard, V. Miele et C. Ambroise. Strategies for online inference of network mixture. 4(2) :687–714.

APPENDIX A

$$
\begin{aligned}
p^i_{N+1} &= \frac{\mathbb{P}(Z_{N+1}=1, X_{i(N+1)}=x_{i(N+1)}, Z_i=z_i,\, \forall i\in\{1,\dots,N\})}{\mathbb{P}(X_{i(N+1)}=x_{i(N+1)}, Z_i=z_i,\, \forall i\in\{1,\dots,N\})}\\
&= \frac{\beta^{\sum_{i:z_i=0} x_{i(N+1)}}(1-\beta)^{(N_{\bar c}-\sum_{i:z_i=0} x_{i(N+1)})}}{\beta^{\sum_{i:z_i=0} x_{i(N+1)}}(1-\beta)^{(N_{\bar c}-\sum_{i:z_i=0} x_{i(N+1)})}}\\
&\times \frac{\gamma\alpha^{d^{in}_{N+1}}(1-\alpha)^{(N_c-d^{in}_{N+1})}}{\gamma\alpha^{d^{in}_{N+1}}(1-\alpha)^{(N_c-d^{in}_{N+1})}+(1-\gamma)\beta^{d^{in}_{N+1}}(1-\beta)^{(N_c-d^{in}_{N+1})}}\\
&= \frac{\gamma\alpha^{d^{in}_{N+1}}(1-\alpha)^{(N_c-d^{in}_{N+1})}}{\gamma\alpha^{d^{in}_{N+1}}(1-\alpha)^{(N_c-d^{in}_{N+1})}+(1-\gamma)\beta^{d^{in}_{N+1}}(1-\beta)^{(N_c-d^{in}_{N+1})}},
\end{aligned}
$$

with $N_{\bar c}=\sum_{i=1}^N(1-z_i)$.

$$
\begin{aligned}
p^{io}_{N+1} &= \frac{\mathbb{P}(Z_{N+1}=1, X_{i(N+1)}=x_{i(N+1)}, X_{(N+1)i}=x_{(N+1)i}, Z_i=z_i,\, \forall i\in\{1,\dots,N\})}{\mathbb{P}(X_{i(N+1)}=x_{i(N+1)}, X_{(N+1)i}=x_{(N+1)i} Z_i=z_i,\, \forall i\in\{1,\dots,N\})}\\
&= \frac{\beta^{\sum_{i:z_i=0}(x_{i(N+1)}+x_{(N+1)i})}(1-\beta)^{(2N_{\bar c}-\sum_{i:z_i=0}(x_{i(N+1)}+x_{(N+1)i})}}{\beta^{\sum_{i:z_i=0}(x_{i(N+1)}+x_{(N+1)i})}(1-\beta)^{(2N_{\bar c}-\sum_{i:z_i=0}(x_{i(N+1)}+x_{(N+1)i})}}\\
&\times \frac{\gamma\alpha^{d_{N+1}}(1-\alpha)^{(2N_c-d_{N+1})}}{\gamma\alpha^{d_{N+1}}(1-\alpha)^{(2N_c-d_{N+1})}+(1-\gamma)\beta^{d_{N+1}}(1-\beta)^{(2N_c-d_{N+1})}}\\
&= \frac{\gamma\alpha^{d_{N+1}}(1-\alpha)^{(2N_c-d_{N+1})}}{\gamma\alpha^{d_{N+1}}(1-\alpha)^{(2N_c-d_{N+1})}+(1-\gamma)\beta^{d_{N+1}}(1-\beta)^{(2N_c-d_{N+1})}}.
\end{aligned}
$$

APPENDIX B

$$
\begin{aligned}
p^i_{N+1} &> s\\
\frac{\gamma\alpha^{d^{in}_{N+1}}(1-\alpha)^{(N_c-d^{in}_{N+1})}}{\gamma\alpha^{d^{in}_{N+1}}(1-\alpha)^{(N_c-d^{in}_{N+1})}+(1-\gamma)\beta^{d^{in}_{N+1}}(1-\beta)^{(N_c-d^{in}_{N+1})}} &> s\\
\frac{\gamma\alpha^{d^{in}_{N+1}}(1-\alpha)^{(N_c-d^{in}_{N+1})}}{(1-\gamma)\beta^{d^{in}_{N+1}}(1-\beta)^{(N_c-d^{in}_{N+1})}} &> \frac{s}{1-s}\\
d^{in}_{N+1}\log\left(\frac{\alpha(1-\beta)}{(1-\alpha)\beta}\right) &> \log\left(\frac{s(1-\beta)^{N_c}(1-\gamma)}{(1-s)(1-\alpha)^{N_c}\gamma}\right)\\
d^{in}_{N+1} &> d^{in}_{min},
\end{aligned}
$$

with

$$d_{min}^{in} = \left\lfloor \frac{\log\left(s(1-\beta)^{N_c}(1-\gamma)\right) - \log\left((1-s)(1-\alpha)^{N_c}\gamma\right)}{\log\left(\alpha(1-\beta)\right) - \log\left((1-\alpha)\beta\right)} \right\rfloor$$