Disentangling the Wikipedia Category Graph for Corpus Extraction

Axel-Cyrille Ngonga Ngomo and Frank Schumacher

***Abstract*—In several areas of research such as knowledge management and natural language processing, domain-specific corpora are required for tasks such as terminology extraction and ontology learning. The presented investigations herein are based on the assumption that Wikipedia can be used for the purpose of corpus extraction. It presents the advantage of possessing a semantic layer, which should ease the extraction of domain-specific corpora. Yet, as the Wikipedia category graph is scale-free, it can not be used as it is for these purposes. In this paper, we propose a novel approach to graph clustering called BorderFlow, which we use and evaluate on the Wikipedia category graph. Additional possible applications of these results in the area of information retrieval are presented.**

***Index Terms*—Natural language processing, local graph clustering, corpus extraction.**

I. Introduction

SEVERAL areas of research (e.g., knowledge management, natural language processing (NLP)) require domain-specific knowledge for tasks such as information retrieval (IR), lexicon extraction and ontology learning. In order to remedy the lack of domain-specific text corpora in certain domains, the Web has been used as supplementary data source. The investigations presented herein are based on the assumption that Wikipedia can provide a good starting point for this task, as it provides high-quality text and is freely accessible. A naive approach to the extraction of domain-specific text corpora from Wikipedia would consist of two steps: selecting the node(s) of the category graph which describe best the data required, and fetching iteratively all related categories (related means here, for example, categories appearing in the same articles or subcategories). Yet, such an approach would fail due to the fact that the Wikipedia category graph (WCG) presents a high degree of connectivity as it is scale-free [12]. An iterative approach would thus select too many if not all categories when iterated sufficiently often, since it would tend to integrate hubs. Furthermore, the WCG does not present an explicit similarity relation between categories, which could be used for the purpose described above. In this paper, we present a novel soft graph clustering approach, BorderFlow, which allows the discovery of clusters of paradigmatically related categories. Consequently, it enables the retrieval of domain-specific corpora, which can be extracted by retrieving all the pages tagged with the categories belonging to a certain domain, i.e., to a certain cluster. It is of great importance that the algorithm is fuzzy, as a category can belong to more than one domain. For example, "graph clustering" can be seen as belonging to mathematics and to computer sciences. BorderFlow allows the online detection (i.e. the detection at runtime) of clusters in large graphs generated by a given seed, making it suitable to be used in several Web2.0 applications such as the instantiation and exploration of taxonomies and ontologies and the generation of novel user interfaces for adaptive IR.

The rest of this paper is organized as follows: the next section presents some related work on graph clustering. In the subsequent section, the theoretical background of BorderFlow is elucidated, including a heuristic guaranteeing short run times on large graphs such as the WCG. Thereafter, the current implementation is described. The results achieved on the WCG and further possible applications of this clustering algorithm in the context of NLP are finally discussed.

II. Related Work

Graph clustering algorithms have been a topic of intense research in the past decade. They try to maximize or minimize a given criterion such as conductance, inter-cluster similarity or silhouette factor [9]. Markov Clustering (MCL) [11] for example tries to maximize the flow within a cluster. It is based on the idea that random walks that visits a cluster are likely not to leave the cluster until they have visited many of its vertices. Using a combination of inflation and expansion operators on all elements of the adjacency matrix, the algorithm generates a partition of the graph vertices.

Another clustering algorithm, which uses global information is the Iterative Conductance Cutting (ICC) algorithm [6]. The underlying idea of the algorithm is to iteratively separate clusters by finding minimal conductance cuts. The algorithm is NP-hard by itself, although it can be made polynomial when using a heuristic based on the eigenvalue of the adjacency matrix.

A popular algorithm in the area of NLP is Pantel's Clustering By Committee (CBC, [10]). It is a two-step algorithm, which first discovers unambiguous cluster centers (so-called committees) by computing sub-clusters in the top-k similarity graph generated out a complete similarity graph. Committees maximize the intra-cluster similarity while minimizing the

Manuscript received February 5, 2009. Manuscript accepted for publication March 20, 2009.

Axel-Cyrille Ngonga Ngomo and Frank Schumacher are with the Department of Computer Science, University of Leipzig, Johannisalle 23, Room 5-22, 04103 Leipzig, Germany; e-mail: ngonga@informatik.uni-leipzig.de

inter-cluster similarity. The elements which do not belong to any committee are subsequently clustered in a fuzzy fashion in the second pass. CBC demands the setting of the parameter k, which can lead to too strict/loose committees and thus to an inadequate clustering of the graph at hand. Nevertheless CBC can be used to cluster large graph, as it reduces in the worst case number of edges of the graph from $O(n^2)$ to $O(kn)$, n being the number of nodes.

Another possible approach to clustering large graphs is to give up deterministic behavior, as implemented by Chinese Whispers (CW) [2]. CW begins by labeling each graph node with its own label. Subsequently, it randomly picks nodes and assigns the predominant label in the environment to them. Although it can be used in real world examples, CW does not converge and can thus lead to an infinite computation time when not controlled by a set of thresholds.

A good overview of further graph clustering algorithms can be found in [4]. In the following, a deterministic and threshold-free algorithm for clustering large graphs is proposed. It maximizes the inner cluster similarity while minimizing the intra-cluster similarity, thus maximizing the silhouette factor.

III. The BorderFlow Algorithm

BorderFlow is a general-purpose graph clustering algorithm. It uses solely local information for clustering and achieves a soft clustering of the input graph. The definition of cluster underlying BorderFlow was proposed by [3]. They state that a cluster is a collection of nodes that have more links between them than links to the outside. When considering a graph as the description of a flow system, Flake et al.'s definition of a cluster implies that a cluster X can be understood as a set of nodes such that the flow within X is maximal while the flow from X to the outside is minimal. The idea behind BorderFlow is to maximize the flow from the border of each cluster to its inner nodes (i.e., the nodes within the cluster) while minimizing the flow from the cluster to the nodes outside of the cluster. In the following, we will specify BorderFlow for weighted directed graphs, as they encompass all other forms of non-complex graphs.

A. Formal Specification

Let G = (V, E, ω) be a weighted directed graph with a set of vertices V, a set of edges E and a weighing function ω, which assigns a positive weight to each edge $e \in E$. In the following, we will assume that non-existing edges are edges e such that $\omega(e) = 0$. Before we describe BorderFlow, we need to define functions on sets of nodes. Let $X \subseteq V$ be a set of nodes. We define the set $i(X)$ of inner nodes of X as:

$$i(X) = \{x \in X | \forall y \in V : \omega(xy) > 0 \rightarrow y \in X\}. \quad (1)$$

The set $b(X)$ of border nodes of X is then

$$b(X) = \{x \in X | \exists y \in V \backslash X : \omega(xy) > 0\}. \quad (2)$$

The set $n(X)$ of direct neighbors of X is defined as

$$n(X) = \{y \in V \backslash X \ |\exists x \in X : \omega(xy) > 0\}. \quad (3)$$

In the example of a cluster depicted in Figure 1, $X = \{3, 4, 5, 6\}$, the set of border nodes of X is $\{3, 5\}$, $\{6, 4\}$ its set of inner nodes and $\{1, 2\}$ its set of direct neighbors.

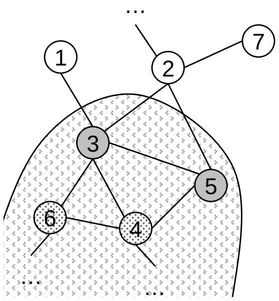


Fig. 1. An exemplary cluster. *The nodes with relief are inner nodes, the grey nodes are border nodes and the white are outer nodes. The graph is undirected.*

Let Ω be the function that assigns the total weight of the edges from a subset of V to another one to these subsets (i.e., the flow between the first and the second subset). Formally:

$$\begin{array}{c} \Omega : 2^V \times 2^V \rightarrow \mathbb{R} \\ \Omega(X, Y) = \sum_{x \in X, y \in Y} \omega(xy). \end{array} \quad (4)$$

We define the border flow ratio $F(X)$ of $X \subseteq V$ as follows:

$$F(X) = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), V\backslash X\big)} = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), n(X)\big)}. \quad (5)$$

Based on the definition of a cluster by [3], we define a cluster X as a node-maximal subset of V that maximizes the ratio $F(X)^1$, i.e.:

$$\forall X' \subseteq V,\ \forall v \notin X : X' = X + v \rightarrow F(X') < F(X). \quad (6)$$

The idea behind BorderFlow is to select elements from the border $n(X)$ of a cluster X iteratively and insert them in X until the border flow ratio $F(X)$ is maximized, i.e., until Equation (6) is satisfied. The selection of the nodes to insert in each iteration is carried out in two steps. In a first step, the set $C(X)$ of candidates $u \in V\backslash X$ which maximize $F(X + u)$ is computed is as follows:

$$C(X) := \underset{u \in n(X)}{\arg\max}\ F(X + u). \quad (7)$$

By carrying out this first selection step, we ensure that each candidate node u which produces a maximal flow to the inside of the cluster X and a minimal flow to the outside of X is selected. The flow from a node $u \in C(X)$ can be divided into three distinct flows:

– the flow $\Omega(u, X)$ to the inside of the cluster,

[1] For the sake of brevity, we shall utilize the notation $X + c$ to denote the addition of a single element c to a set X. Furthermore singletons will be denoted by the element they contain, i.e., $\{v\} \equiv v$.

- the flow $\Omega(u, n(X))$ to the neighbors of the cluster and
- the flow $\Omega(u, V \backslash (X \cup n(X)))$ to the rest of the graph.

Prospective cluster members are elements of $n(X)$. To ensure that the inner flow within the cluster is maximized in the future, a second selection step is necessary. During this second selection step, BorderFlow picks the candidates $u \in C(X)$ which maximize the flow $\Omega(u, n(X))$. The final set of candidates $C_f(X)$ is then

$$C_f(X) := \underset{u \in C(X)}{\arg\max}\ \Omega(u, n(X)). \tag{8}$$

All elements of $C_f(X)$ are then inserted in X if the condition

$$F(X \cup C_f(X)) \geq F(X) \tag{9}$$

is satisfied.

B. A Heuristics for Maximizing the Border Flow Ratio

One drawback of the method proposed above is that it demands the simulation of the inclusion of each node in $n(X)$ in the cluster X before choosing the best ones. Such an implementation can be time-consuming as nodes in terminology graphs can have a high number of neighbors. The need is for a computationally less expensive criterion for selecting a nearly optimal node to optimize $F(X)$. Let us assume that X is large enough. This assumption implies that the flow from the cluster boundary to the rest of the graph is altered insignificantly when adding a node to the cluster. Under this condition, the following two approximations hold:

$$\begin{array}{l} \Omega(b(X), n(X)) \approx \Omega(b(X+v), n(X+v)), \\ \Omega(b(X), v) - \Omega(d(X,v), X+v) \approx \Omega(b(X), v). \end{array} \tag{10}$$

Consequently, the following approximation holds:

$$\Delta F(X, v) \approx \frac{\Omega(b(X), v)}{\Omega(b(X+v), n(X+v))}. \tag{11}$$

Under this assumption, one can show that the nodes that maximize $F(X)$ maximize the following:

$$f(X, v) = \frac{\Omega(b(X), v)}{\Omega(v, V \backslash X)} \text{ for symmetrical graphs.} \tag{12}$$

Now, BorderFlow can be implemented in a two-step greedy fashion by ordering all nodes $v \in n(X)$ according to $1/f(X, v)$ (to avoid dividing by 0) and choosing the node v that minimizes $1/f(X, v)$. Using this heuristic, BorderFlow is easy to implement and fast to run. The resulting main routine is shown in Algorithm 1.

C. Generating Cluster Hierarchies

The resulting clusters can be seen as nodes of a higher-level weighted graph $\Gamma = (\Psi, \Sigma, \chi)$ with Ψ being the set of all generated clusters, Σ being the set of edges between clusters and χ being the relative flow between clusters. Given two clusters X and X', the weight of the edge XX' would then be

```
Data: Graph to cluster
Result: Fuzzy clustering
for each $v \in V$ do
    $X := \{v\}$;
    while $|n(X)| > 0$ do
        $C(X) := \underset{u \in n(X)}{\arg\min}\ 1/f(X, u)$;
        if ($|C(X)| == 1$ && $F(X \cup C(X)) \geq F(X)$)
        then
            $X := X \cup C(X)$;
        else
            $C_f(X) := \underset{u \in C(X)}{\arg\max}\ \Omega(u, n(X))$;
            if ($F(X \cup C_f(X)) \geq F(X)$) then
                $X := X \cup C_f(X)$;
            else
                break;
            end
        end
    end
    store $X$;
end
merge all identical $X$;
return;
```

Algorithm 1. Current implementation of BorderFlow

$$\chi(X, X') = \begin{cases} 0 & \text{if } \Omega(X, V) = 0; \\ \frac{\Omega(X, X')}{\Omega(X, V)} & \text{else.} \end{cases} \tag{13}$$

Using this simple equation, a hierarchical fuzzy clustering of the graph G can be generated in a bootstrapping fashion.

IV. Experimental Setup

Our global aim was to cluster similar nodes to soft clusters (i.e., allowing nodes to occur in more than one cluster). As with all clustering approaches, the need was for a definition of the similarity of the nodes to cluster, i.e., the categories. The graph at hand allows several definitions of similarity, since several relations can exist between categories. We considered the three relations:

- parent-of: parent-of(ζ, ζ') was considered to hold when ζ' was a subcategory of ζ.
- son-of: This relation is the inverse of parent-of.
- shared-article: Categories were considered to be in this relation when they appeared in the same articles.

As the similarity between categories is of paradigmatic nature, we used the sets

$$R(\zeta, r) = \{y : r(x, y)\} \tag{14}$$

to compare the categories ζ. The similarity $\sigma(\zeta, \zeta')$ of two categories ζ and ζ' was computed using the Jaccard metric:

$$\sigma_r(\zeta, \zeta') = \frac{2|R(\zeta, r) \cap R(\zeta', r)|}{|R(\zeta, r) \cup R(\zeta', r)|}. \tag{15}$$

Since we were not interested in assigning hubs (i.e. highly polysemic categories) to the leafs of the clustering hierarchy, we used the connectivity of nodes as a hint for their specificity and clustered only those nodes, which displayed a connectivity below the average connectivity of the graph. In the graph at hand[2], the average connectivity was 295 for “parent-of”, 8 for “son-of” and 60 for “shared-article”. Nevertheless, nodes with a connectivity above average could be included in clusters.

V. Results

Table I displays statistics on the graphs used for clustering. They were 244,545 initial categories. As a high percentage of the categories available do not have any descendant, clustering over son-of covered solely 31.63% of the categories available. The other two relations covered approximately the same percentage of categories (82.21% for shared-article, 82.07 for parent-of, see Table I). In order to evaluate the clustering quality achieved by BorderFlow, we adapted the silhouette value to graph and measure the value $\sigma(C)$ for the clusters C generated as follows:

$$\sigma(C) = \frac{\Omega'(C,C) - \Omega'(C,n(C))}{max\{\Omega'(C,C), \Omega'(C,n(C))\}}, \tag{16}$$

where

$$\Omega'(X,Y) = \frac{|\{xy : x \in X \wedge y \in Y| \wedge \omega(xy) > 0}{\sum_{x \in X, y \in Y} \frac{1}{\omega(xy)}}, \tag{17}$$

a value of 1 hinting toward a good clustering and -1 toward an unsuitable clustering. For reasons of brevity, $\sigma(C)$ will be henceforth called silhouette value.

TABLE I
Statistics on Cluster Extraction (1)

Relation	shared-article	son-of	parent-of
Categories	201,049	77,292	200,688
Clusters	93,331	28,568	90,418
Avg. N/C	3.59	2.29	8.63
Avg. C/N	7.74	6.20	19.15
Coverage	82.21%	31.61%	82.07%

Figure 2 shows the distribution of the silhouette over all the clusters computed on the three graphs. The best clustering was achieved on the *shared-article*-graph (see Figure 2(a)). We obtained the highest mean (0.92) with the smallest standard deviation (0.09). An analysis of the silhouettes of the clusters computed by using the *parent-of*-graph revealed that the mean of the silhouette lied around 0.74 with a standard deviation of 0.24 (see Figure 2(b)). The smaller average silhouette value was mainly due to the high connectivity of the similarity graph generated using this relation, resulting into large clusters and thus a higher flow to the outside (see Table II). Clustering the *child-of*-graph yielded the worst results, with a mean of 0.20 and a standard deviation of 0.19. Figure 3 shows an example of a cluster containing “Computational Linguistics”.

The high average number of clusters per node (i.e., the number of cluster to which a given node belongs) show how polysemic the categories contained in the WCG are. By using the clustering resulting from our experiments with the *shared-article* relation, one can subdivide the WCG into domain-specific categories and use these to extract domain-specific corpora from Wikipedia. Furthermore, BorderFlow allows the rapid identification of categories similar to given seed categories. Thus, BorderFlow can be used for other NLP applications such as query expansion [1], topic extraction [7] and terminology expansion [5].

TABLE II
Statistics on Cluster Extraction (2)

Relation	Mean	Standard deviation
shared-article	0.92	0.09
parent-of	0.74	0.24
son-of	0.20	0.19

From a qualitative point of view, the clusters generated by BorderFlow on the WCG differ heavily depending on the relation chosen. Examples of clusters around “Computational Linguistics” are shown in Fig 3. Note that the cluster generated using “parent-of” is not shown in its completeness, as it contains 52 categores.

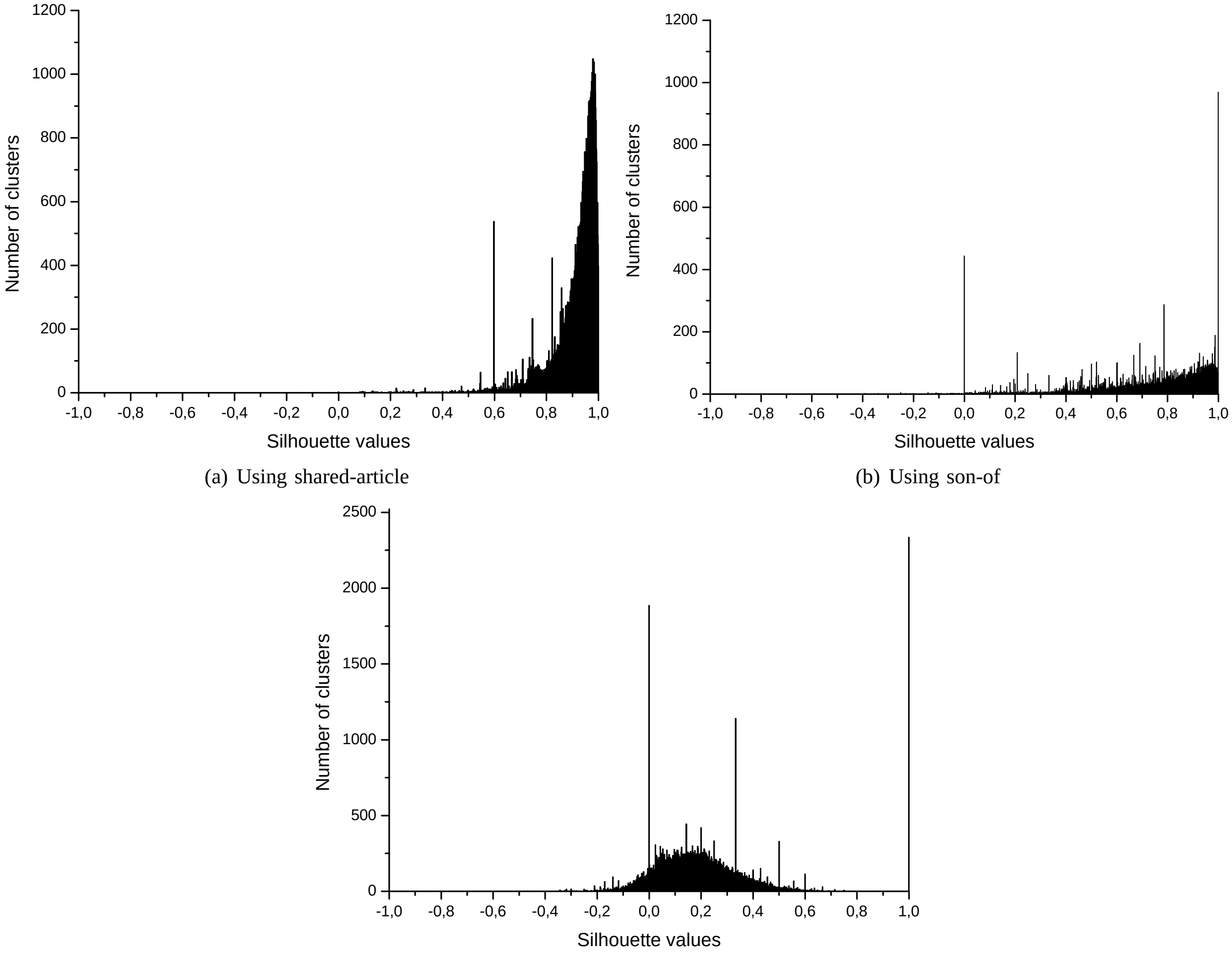
Our results support the idea that polysemantic categories should belong to many clusters. Hence, it supports the generation of domain-specific views on categories. Table III shows the example of clusters, which contain “Computational Linguistics”.

VI. Further Applications

In the case of the WCG, BorderFlow generates classes of similar categories, which can be used to automatically discover new sub-domains or refinements of existing domains in Wikipedia. BorderFlow can yet be used for several other purposes in NLP-related areas such as concept retrieval, IR, etc.

As classifications, taxonomies and similar graph structures can present a large number of nodes (e.g. classes, instances etc.), searching through them can be a very tedious process. As shown when clustering the WCG, the definition of similarity can be based on different relations. As BorderFlow allows for clustering at runtime, the different similarity definitions make it possible to generate different facets of a node given a graph and the relations implemented in the latter. Figure 3 shows an example of different facets of “Computational Linguistic”. By these means, several views on a given node can be generated using BorderFlow, allowing a more efficient exploration of large-scale structures. This approach can be used to browse through large domain-specific ontologies or to implement a browsing approach to information retrieval, using collocation networks, term nets, document nets or similar graphs for the browsing layer [8].

[2] English version of the WCG, version of July 2007

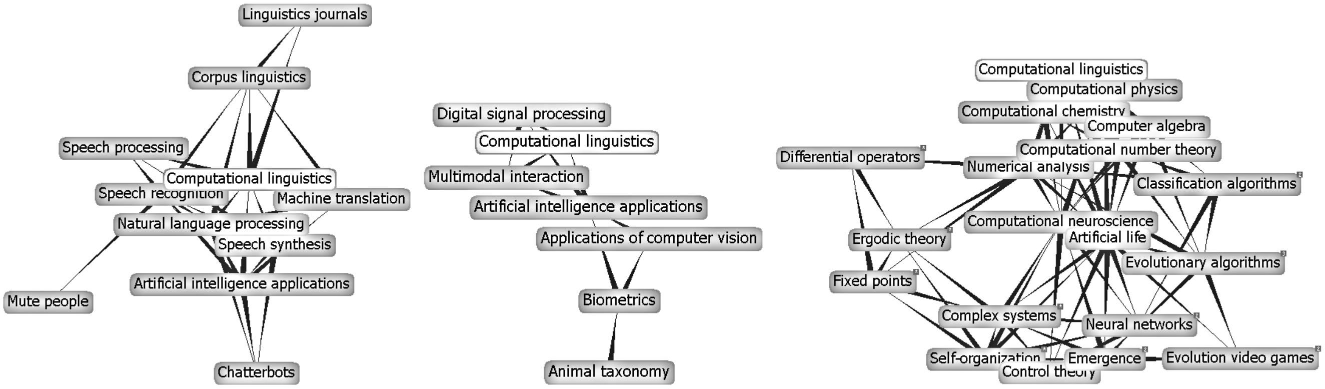


(a) Using shared-article

(b) Using son-of

(c) Using parent-of

Fig. 2. Distribution of silhouette values



(a) Using shared-article

(b) Using son-of

(c) Using parent-of

Fig. 3. Examples of clusters containing “Computational Linguistics”

VII. Conclusion

We presented and a new clustering algorithm and evaluated it by clustering the WCG. The clustering reveals that BorderFlow allows the retrieval of facets of categories. Based on these results, a hierarchical structure can be generated out of the WCG, allowing for an extraction of domain-specific sub-corpora. The clustered category graph can be used for several other purposes. It allows the implementation of several

TABLE III
Clusters Containing “Computational linguistics” Generated using the “parent-of” Relation

Related scientifical disciplines
Logic, Applied and interdisciplinary physics, Scientific modeling, Astrology, Museology, Systems theory, Cybernetics, Library and information science, Urban studies and planning, Artificial intelligence, Cognitive science, Neuroscience, Biomedical engineering, Bioinformatics, Neural engineering, Prosthetics, Evolutionary psychology, *Computational linguistics*, Cultural studies, Semiotics, Political economy, Ethnic studies, Human sciences, Science studies, Mathematical physics, Science and technology studies, Transdisciplinarity

Linguistics
Abbreviations, Words, Linguists, Terminology, Linguistic morphology, Toponymy, Language comparison, Linguistic typology, Ethnopoetics, Psycholinguistics, Countries by language, Graphemes, Phonology, Translation, Philosophy of language, Sociolinguistics, Buzzwords, Romanization, Speech impediments, *Computational linguistics*, Language education, Phonetics, Language games, Structuralism, Profanity, Syntax, Corpus linguistics, Transliteration, Semantics, Rhetoric, Dialectology, Lexicology, Lexicography, Recording, Discourse analysis, Puns, Linguistics lists, Onomastics, Historical linguistics, Euphemisms, Evaluation of machine translation, Metaphors, Onomatopoeia, Grammar, Word games, Generative linguistics, Philology, Indo-European linguistics, Interlinguistics, Lexis, Linguistics journals, Sprachbund, Language and mysticism, Language regulators, Linguistic societies, Linguistics books, Swadesh lists, Misnomers, Word coinage, Neurolinguistics, Proper nouns, Pragmatics, Semitic linguistics, Vowel shifts, Word play, Zero (linguistics)

Related theories
Computer algebra, Classification algorithms, Computational number theory, Evolutionary algorithms, Numerical analysis, Self-organization, Artificial life, Artificial Life models, Emergence, Evolution video games, Bayesian networks, Free Bayesian statistics software, Bifurcation theory, Fixed points, Creatures, Boundary conditions, Partial differential equations, Morse theory, Chaos theory, Chaotic maps, Non-linear systems, Hamiltonian mechanics, Clathrates, Complex systems, Computational chemistry, *Computational linguistics*, Computational neuroscience, Neural networks, Computational physics, Machine learning researchers, Control theory, Diffeomorphisms, Differential operators, Numerical differential equations, Ordinary differential equations, Stability theory, Stochastic differential equations, Waves, Differential forms, Foliations, Symplectic topology, Queueing theory, Entropy and information, Ergodic theory, Limit sets, Non-equilibrium thermodynamics, Random dynamical systems, Free econometrics software, Free plotting software, Free web analytics software, Vector calculus, Subharmonic functions

retrieval techniques such as browsing and berry-picking (see [1], [8]) on Wikipedia . Furthermore, by simplifying the graph structure, it allows to build hierarchies that ease the use of reasoners and further logics oriented tools on the graph.

From an algorithmic point of view, BorderFlow allows a fast deterministic clustering of complex graphs of all sort, since it uses solely local information. When using the heuristics proposed, it is highly scalable. Furthermore, it can also be used on directed graphs allowing negative edges weights without further alteration of the formalisms described above. The soft clustering of the graph generated by BorderFlow allows a hierarchical clustering of the result in a bootstrapping fashion, since the results can be seen as a higher level graph. Out of the results presented in this paper, we intend to generate a fuzzy cluster hierarchy, which will allow an hierarchical view on the domains contained in the the Wikipedia graph. By these means, we will generate a new view on the graph, not replacing the existing relations but rather completing them by adding the paradigmatic relatedness of categories.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press/Addison-Wesley Longman Publishing Co., Harlow, England, 1999.

[2] C. Biemann. Chinese whispers - an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the HLT-NAACL-06 Workshop on Textgraphs-06*, New York, USA, 2006.

[3] G. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–160, Boston, MA, August 20–23 2000.

[4] G. Gan, C. Ma, and J. Wu. *Data Clustering: Theory, Algorithms, and Applications (ASA-SIAM Series on Statistics and Applied Probability)*. SIAM, 2007.

[5] C. Jacquemin, J. Klavans, and E. Tzoukermann. Expansion of multi-word terms for indexing and retrieval using morphology and syntax. In *Proceeding of 35th ACL*, pages 24–31, 1997.

[6] R. Kannan, S. Vampala, and A. Vetta. On clustering: good, bad and spectral. In *Proceedings of 41st Annual Symposium on Foundations of Computer Science*, pages 367–378, New York, USA, 2000.

[7] A. Maguitman, D. Leake, T. Reichherzer, and F. Menczer. Dynamic extraction topic descriptors and discriminators: towards automatic context-based topic search. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 463–472, New York, NY, USA, 2004. ACM.

[8] A.-C. Ngonga Ngomo and F. Schumacher. Involving the user in semantic search. In Michael J. Smith and Gavriel Salvendy, editors, *HCI (8)*, volume 4557 of *Lecture Notes in Computer Science*, pages 507–516. Springer, 2007.

[9] V. Kumar P.-N. Tan, M. Steinbach. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[10] P. Pantel. *Clustering by Committee*. PhD thesis, University of Alberta, Edmonton, Alberta, Canada, 2003.

[11] S. van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, 2000.

[12] T. Zesch and I. Gurevych. Analysis of the Wikipedia Category Graph for NLP Applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT 2007)*, pages 1–8, 2007.