Université de Nice - Sophia Antipolis – UFR Sciences
École Doctorale STIC

THÈSE

Présentée pour obtenir le titre de :

Docteur en Sciences de l'Université de Nice - Sophia Antipolis

Spécialité : INFORMATIQUE

par

Marina SOKOL

Équipe d'accueil : Maestro – INRIA Sophia Antipolis

GRAPH-BASED SEMI-SUPERVISED LEARNING METHODS AND QUICK DETECTION OF CENTRAL NODES

Thèse dirigée par Paulo GONÇALVES et Philippe NAIN

Soutenance à l'Inria le 29 Avril 2014, à 14:00 devant le jury composé de :

Président :	Walid	DABBOUS	Inria Sophia Antipolis
Directeurs :	Paulo	GONÇALVES	Inria Rhône-Alpes and ENS Lyon
	Philippe	NAIN	Inria Sophia Antipolis
Rapporteurs :	Ravi	KUMAR	Google Research
	Stefano	LEONARDI	Sapienza University of Rome
	Hichem	SAHBI	LTCI, CNRS, TELECOM ParisTech

Thèse

Méthodes d'Apprentissage Semi-Supervisé basé sur les Graphes et Détection Rapide des Noeuds Centraux

Marina Sokol

Avril 2014

Graph-based Semi-supervised Learning Methods and Quick Detection of Central Nodes

by
Marina Sokol
Thesis advisors: Paulo Gonçalves and Philippe Nain
Maestro team, Inria Sophia Antipolis, France

Abstract

Semi-supervised learning methods constitute a category of machine learning methods which use labelled points together with unlabelled data to tune the classifier. The main idea of the semi-supervised methods is based on an assumption that the classification function should change smoothly over a similarity graph, which represents relations among data points. This idea can be expressed using kernels on graphs such as graph Laplacian. Different semi-supervised learning methods have different kernels which reflect how the underlying similarity graph influences the classification results. In the first part of the thesis, we propose a generalized optimization approach for the graph-based semi-supervised learning which implies as particular cases the Standard Laplacian, Normalized Laplacian and PageRank based methods and extends to new ones, through the introduction of a free parameter. Using random walk theory, we provide insights about the differences among the graph-based semi-supervised learning methods and give recommendations for the choice of the kernel parameters and labelled points. In particular, it appears that it is preferable to choose a kernel based on the properties of the labelled points. We have also characterized the limiting behaviour of the methods with respect to the regularization parameter. We show that the PageRank based method is the only method among the methods of the considered family which shows robustness when the classes are unbalanced. We have illustrated all theoretical results with the help of synthetic and real data. As one example of real data we consider classification of content and users in P2P systems. This application demonstrates that the proposed family of methods scales very well with the volume of data. Then, the second part of the thesis is devoted to quick detection of network central nodes. The algorithms developed in the second part of the thesis can be applied for the selection of quality labelled data but also have other applications in information retrieval. Specifically, we propose random walk based algorithms for quick detection of large degree nodes and nodes with large values of Personalized PageRank. We prove that on configuration type networks our algorithm finds top degree nodes with high probability in sublinear time. We develop stopping criteria which require very little knowledge of the network topology. Finally, in the end of the thesis we suggest new centrality measure, which generalizes both the current flow betweenness centrality and PageRank. This new measure is particularly well suited for detection of network vulnerability. For working with large volumes of real data, like P2P data, we needed to write a java library, which is described in a special section and can be useful for other researchers.

ACKNOWLEDGMENTS

I would like to thank my thesis advisors Paulo Gonçalves and Philippe Nain for their generous guidance and many helpful advices. I thank very much Konstantin Avrachenkov for inspiring collaboration and emotional support. I also would like to thank Alcatel-Lucent for providing scholarship and Ludovic Noirie from Alcatel-Lucent for motivating discussions. I am very grateful to the thesis reviewers: Ravi Kumar, Stefano Leonardi and Hichem Sahbi, whose comments and recommendations help to improve the presentation of the work. Also, I would like to thank a lot Maestro team members and in particular Nicaise Choungmo Fofack, Laurie Vermeersch and Valeria Neglia. Last but not least, I would like to thank from the bottom of my heart my dad, Mikhail Sokol.

Marina Sokol
Marina.Sokol@inria.fr
Inria Sophia Antipolis, France

Dedication

To my mother Tatiana Sokol.

CONTENTS

Abstract iii

Acknowledgements v

Figures xiv

Tables xvi

1 Introduction and summary of main results 3

I Graph-based Semi-supervised Learning Methods 9

2 Optimization framework for semi-supervised learning methods 11
- 2.1 Introduction and summary of the results . . . 11
- 2.2 Generalized Optimization Framework . . . 13
- 2.3 Experiments . . . 19
  - 2.3.1 Les Miserables example . . . 19
  - 2.3.2 Wikipedia-math example . . . 19
- 2.4 Conclusions . . . 23

3 Random walk approach for semi-supervised learning methods 27
- 3.1 Introduction and summary of the results . . . 27
- 3.2 General theoretical considerations . . . 29
- 3.3 Evaluation . . . 32
- 3.4 Conclusions and general recommendations . . . 37

4 Semi-supervised methods for P2P content and user classification 39
- 4.1 Introduction and summary of the results . . . 39
- 4.2 Datasets and method implementation description . . . 41
- 4.3 Results of classification of content and users . . . 44

4.4 Classification of Video plus Music subgraph 46
4.5 Classification of untagged content 47
4.6 The effect of σ and α 49
4.7 Conclusions 53

II Quick Detection of Central Nodes in Complex Networks 55

5 Quick detection of nodes with large degrees 57
5.1 Introduction and summary of the results 57
5.2 Random walk with uniform jumps 59
5.3 Estimating the largest degrees in the configuration network model 66
5.4 Stopping criteria 69
5.5 Relaxation of top k lists 72
5.6 Conclusions 74

6 Quick detection of nodes with large values of PageRank 77
6.1 Introduction 77
6.2 Monte Carlo methods 79
6.3 Variance based performance comparison and CLT approximations 83
6.4 Convergence based on order 86
6.5 Solution relaxation 88
6.6 Conclusions 91

7 Alpha current flow centrality 93
7.1 Introduction and summary of the results 93
7.2 Alpha current flow betweenness 95
7.3 Computation of α-CF betweenness 98
7.4 Truncated α-CF betweenness 99
7.5 Datasets 101
7.6 Numerical results for α-CF betweenness 102
7.7 Centrality measures and network vulnerability 105
7.8 Conclusions 107

8 Appendix: Software description 109
8.1 Summary 109
8.2 Description 109
8.2.1 The formats 109
8.2.2 The namers 110

8.2.3 The algorithms 110
8.2.4 The expert/seeds files 111
8.2.5 The estimations 112
8.2.6 How to write your own graph implementation 112

9 Conclusions and Future Research **115**

10 Résumé en Français **119**

Bibliography **120**

FIGURES

1.1 A link between two contents. 4

2.1 Fitting and smoothness terms. 17
2.2 Les Miserables example. 20
2.3 Wikipedia-math example: Modularity and Precision of the classification. 21
2.4 Wikipedia-math example: Modularity of the classification for different number of labels. 24
2.5 Wikipedia-math example: Precision of the classification for different number of labels. 25

3.1 Clustered Preferential Attachment Model: Precision of the classification. 35
3.2 Characteristic network model. 36
3.3 Handwritten digits: Precision of the classification. 36

4.1 Content P2P Graph: Precision of the classification. 51
4.2 User P2P Graph: Precision of the classification. 52

5.1 Histograms of hitting times in the PA network. 60
5.2 Average number of correctly detected elements in top-10 for PA. 72
5.3 Average number of correctly detected elements in top-10 for UK. 73

6.1 The number of correctly detected elements by MC End Point for seed nodes with the same name. 81
6.2 The number of correctly detected elements for seed node Michael Jackson. 82
6.3 The number of correctly detected elements by MC End Point for two Web pages. . 82

7.1 The number of pairs s, t with $x^{(s,t)}_{(v,w)} > x$ over all pairs (s, t) (solid line) and only pairs with $v, w \neq s$ (dashed line). 99
7.2 Correlations between α-CF betweenness and truncated α-CF betweenness with CF-betweenness as a function of α. 102

7.3 Distribution of α-CF betweenness scores in the Enron graph, truncated (dashed line) and not truncated (solid line). On the x-axis are the values of α-CF betweenness, on the y-axis the number of edges/nodes with the score larger than x. 103
7.4 Inverse average distance as a function of the fraction of removed top-nodes according to different betweenness centrality measures. 105
7.5 The size of the largest connected component as a function of the fraction of removed top-nodes according to different betweenness centrality measures. . . . 106

TABLES

3.1 Comparison between different methods in terms of classification errors 34

4.1 The content graphs after preprocessing. 41
4.2 The quantity of language base line expert classifications. 42
4.3 The quantity of topic base line expert classifications. 42
4.4 Accuracy of the classifications for the $g(2,10)$ dataset by languages. 45
4.5 Accuracy of the classifications for the $g(2,10)$ dataset by topics. 45
4.6 Accuracy of the classifications for the user dataset by languages. 45
4.7 Accuracy of the classifications for the user dataset by topics. 45
4.8 Accuracy and Cross-Validation (CV) matrix for music&audio vs video&movies classification, $\alpha = 0.5$. 46
4.9 Accuracy for "Other Video" subgraph classification, $\alpha = 0.5$. 48
4.10 Cross-Validation matrix for "Other Video" subgraph classification, TopPR 10 labeled points, $\alpha = 0.5$. 48
4.11 Cross-Validation matrix for "Other Video" subgraph classification, TopPR 15 labelled points, $\alpha = 0.5$. 48
4.12 Statistics for accuracy for "Other Video" subgraph classification, $\alpha = 0.5$, random labeled points, 100 experiments. 48
4.13 Cross-Validation matrix for the user graph classification by languages, $\alpha = 0.1$, precision 83.71%. 50
4.14 Cross-Validation matrix for the user graph classification by languages, $\alpha = 0.95$, precision 68.92%. 50
4.15 Cross-Validation matrix for the user graph classification by languages, $\alpha = 0.999$, Precision 86.65%. 50

6.1 MC End Point for the Jim Jackson Web page: means and Standard Deviations. . . 85

7.1 Datasets characteristics. 101
7.2 Kendall tau for centrality measures in Dolphin social network. 103

7.3 Kendall tau for centrality measures in the social graph VKontakte AMCP. 104
7.4 Kendall tau for centrality measures in the Watts-Strogatz graph (n=1000, k=12, p=0.150). 104

Publications of Marina Sokol:

Articles in journals:

- K. Avrachenkov, N. Litvak, M. Sokol and D. Towsley, "Quick detection of nodes with large degrees", Accepted to *Internet Mathematics* journal. To appear in 2014.

Articles in refereed conferences and workshops:

- K. Avrachenkov, P. Gonçalves and M. Sokol, "On the choice of kernel and labelled data in semi-supervised learning methods", In the 10th International Workshop on Algorithms and Models for the Web Graph, WAW 2013.
- K. Avrachenkov, N. Litvak, V. Medyanikov and M. Sokol, "Alpha current flow betweenness centrality", In the 10th International Workshop on Algorithms and Models for the Web Graph, WAW 2013.
- K. Avrachenkov, P. Gonçalves, A. Legout and M. Sokol, "Classification of content and users in BitTorrent by semi-supervised learning methods", In the IEEE IWCMC International Workshop on Traffic Analysis and Characterization, TRAC 2012 (**Best Paper Award**).
- K. Avrachenkov, N. Litvak, M. Sokol and D. Towsley, "Quick detection of nodes with large degrees", In the 9th International Workshop on Algorithms and Models for the Web Graph, WAW 2012.
- K. Avrachenkov, P. Gonçalves, A. Mishenin and M. Sokol, "Generalized optimization framework for graph-based semi-supervised learning", In the SIAM conference on Data Mining, SDM 2012.
- K. Avrachenkov, P. Gonçalves, A. Legout and M. Sokol, "Graph based classification of content and users in BitTorrent", In the NIPS Big Learning Workshop, 2011.
- K. Avrachenkov, N. Litvak, D. Nemirovsky, E. Smirnova and M. Sokol, "Quick detection for top-k Personalized PageRank lists", In the 8th International Workshop on Algorithms and Models for the Web Graph, WAW 2011.

1

INTRODUCTION AND SUMMARY OF MAIN RESULTS

A recent patent [78] originated from Facebook proposes a method for detection of unauthorized content published by users of a social networking system. The method is in particular based on the analysis of social ties. One can state the problem in a broader context. How to use social/user ties to classify content according to some qualities. Examples of content qualities are copyright infringement, content language, media type (e.g., audio, photo, video), content type (e.g., video movie, video music clip, video lecture) and sub-types video drama movie, video comic movie, video teleseries etc. Such classification can help not only to fight the distribution of content with copyright violation but also to help the design of next generation search engines and content aware networking.

To be more specific, let us start with a high level description of the content classification in Peer-to-Peer (P2P) systems developed in the present thesis. Suppose we would like to classify the content according to languages. Of course, the obvious thing that comes to our mind is to use the title of the content. However, very often the language of the title is misleading. A title can be spelled the same way in several European languages. For instance the movie "Inception" has the same title in English and in French. Even if a movie is not in English language, it can be tagged with an English translation of its original title. This can be explained by the fact that some languages, like the Russian language, have non-latin alphabet and a user who tags the content prefers to use the latin keyboard and an English language translation of the title. Many Asian languages use logograms and it is much easier for the user to tag the content in English translation or transliteration. As in the patent [78], we intend to use ties between users and also between users and content to classify content and users. For instance, we establish a

semantic link between two contents if these two contents are downloaded by at least one same user (see Figure 1.1). A rationale behind such link creation process is that a user typically has preference for a certain type of content and watches movies / read books mostly in his/her native language. Thus, this way we can create a similarity or semantic graph for P2P content and its respective adjacency matrix.

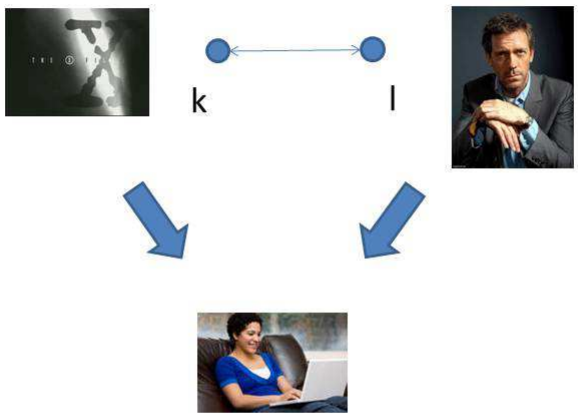


Figure 1.1: A link between two contents.

To classify P2P content and users, we propose to use the framework of the graph-based semi-supervised learning. There are several excellent books and surveys devoted to the semi-supervised learning, see e.g. [83, 85, 33, 4]. Let us recall the main principal ideas of the semi-supervised and, in particular, *graph-based* semi-supervised learning technique. Typically, the labelled data is rare and expensive to obtain. This is because one needs to organize the work of several human experts and to use special devices and sensors. On contrary, nowadays a huge amount of unlabelled data is available to us (effect of "Big Data"). The main idea of the semi-supervised learning approach is to make a synergy between the labelled and unlabelled data. In the context of the graph-based semi-supervised learning this is possible if a so-called *smoothness assumption* or *label consistency assumption* takes place. This assumption says that if two nodes of the similarity graph are connected by many weighted shortest paths, the data points represented by those two nodes should very likely belong to the same class.

Denote by W the weighted adjacency (similarity) matrix representing relations among the data points. Let N be the total number of data points, including the labelled points. In some applications the similarity graph is naturally provided by the application in question (e.g., P2P graphs, Hyper-text graphs, graph of social connections). In other applications, the data points are represented by vectors of attributes (X_i, $i = 1, ..., N$) and the weights of the similarity matrix can be calculated for instance using *Radial Basis Function (RBF)*

$$w_{ij} = \exp(-\|X_i - X_j\|^2/\gamma),$$

or the adjacency matrix can be constructed using the k-Nearest Neighbors (kNN) method (see e.g., [33]). Here we assume that W is symmetric and the underlying graph is connected. Denote by D a diagonal matrix with its (i,i)-element equal to the sum of the i-th row of matrix W: $d_i = \sum_{j=1}^{N} w_{ij}$.

Suppose we need to classify data points into K classes and assume P data points are labelled. That is, we know the class to which each labelled point belongs. Denote by V_k, the set of labelled points in class $k = 1, ..., K$. Thus, $|V_1| + ... + |V_K| = P$.

Define an $N \times K$ matrix Y as

$$Y_{ik} = \begin{cases} 1, & \text{if } i \in V_k, \text{ i.e., point } i \text{ is labelled as a class } k \text{ point,} \\ 0, & \text{otherwise.} \end{cases}$$

We refer to each column Y_{*k} of matrix Y as a labeling function. Also define an $N \times K$ matrix F and call its columns F_{*k} classification functions. The general idea of the graph-based semi-supervised learning is to find classification functions so that on the one hand they will be close to the corresponding labeling function and on the other hand they will change smoothly over the graph associated with the similarity matrix. This general idea can be expressed with the help of optimization formulation. In particular, there are two widely used optimization formulations. The first formulation, the Standard Laplacian based formulation [82], is as follows:

$$\min_F \{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\|F_{i*} - F_{j*}\|^2 + \mu \sum_{i=1}^{N} d_i\|F_{i*} - Y_{i*}\|^2\}, \tag{1.1}$$

and the second, the Normalized Laplacian based formulation [81], is as follows:

$$\min_F \{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\|\frac{F_{i*}}{\sqrt{d_i}} - \frac{F_{j*}}{\sqrt{d_j}}\|^2 + \mu \sum_{i=1}^{N} \|F_{i*} - Y_{i*}\|^2\}, \tag{1.2}$$

where μ is a regularization parameter. In fact, the parameter μ represents a trade-off between the closeness of the classification function to the labeling function and its smoothness over the graph.

In Chapter 2 we construct a *generalized optimization framework*, which has as particular cases the two above mentioned formulations. Namely, we suggest the following optimization criterion

$$\min_{F}\{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\|{d_i}^{\sigma-1}F_{i*} - {d_j}^{\sigma-1}F_{j*}\|^2 + \mu\sum_{i=1}^{N}{d_i}^{2\sigma-1}\|F_{i*} - Y_{i*}\|^2\}. \tag{1.3}$$

The introduction of a new parameter σ gives more flexibility in tuning the criterion. In addition to the Standard Laplacian formulation ($\sigma = 1$) and the Normalized Laplacian formulation ($\sigma = 1/2$), we obtain the third very interesting case when $\sigma = 0$. We show in Chapter 2 that this particular case corresponds to PageRank based clustering [10].

In Chapter 2 we use power series to analyse the important limiting case when $\mu \to 0$. The asymptotic analysis shows that only the PageRank based method ($\sigma = 0$) produces stable classification in the case of unbalanced data. Then, we provide initial intuition about the effects from the choice of various values of parameter σ. We illustrate theoretical considerations with several numerical examples.

In Chapter 3 we continue to investigate the effect of the choice of σ and labelled data on the results of classification. We make extensive use of the theory of random walks on graphs. In particular, in our context it is helpful to consider a random walk with absorption $\{S_t \in \{1, ..., N\}$, $t = 0, 1, ...\}$. At each step with probability $\alpha = 2/(2 + \mu)$ the random walk chooses next node among its neighbours uniformly and with probability $1 - \alpha$ goes into the absorbing state. The probabilities of visiting nodes before absorption given the random walk starts at node j, $S_0 = j$, are provided by the distribution

$$\mathrm{ppr}(j) = (1 - \alpha)e_j^T\left(I - \alpha D^{-1}W\right)^{-1}, \tag{1.4}$$

which is the Personalized PageRank vector with respect to seed node j [45] (the Personalized PageRank will actually be one of the recurring topics of the thesis). Here e_j denotes the j-th element of the standard basis. The central result of Chapter 3 is the following theorem.

Theorem 1.1 *Data point* i *is classified by the generalized semi-supervised learning method* (1.3) *into class* k*, if*

$$\sum_{p\in V_k} d_p^{\sigma} q_{pi} > \sum_{s\in V_{k'}} d_s^{\sigma} q_{si}, \quad \forall k' \neq k, \tag{1.5}$$

where q_{pi} *is the probability of reaching state* i *by the random walk before absorption if* $S_0 = p$.

Theorem 1.1 has several important implications. First, it is very interesting to observe that, using (1.5), one can decouple the effects from the choice of α (or μ) and σ. A change in the value of α (or μ) only influences the factor q_{pi} and a change in the value of σ only affects the factor d_p^{σ}. Second, if $\sigma = 0$ and $|V_k| = \mathrm{const}(k)$, one can expect that smaller classes will attract a larger number of "border points" than larger classes. This effect, if needed, can be

compensated by increasing σ away from zero. And third, if labelled points have the same degree ($d_p = d$, $p \in V_k$, $k = 1, ..., K$), all considered semi-supervised learning methods provide the same classification. These and other implications are discussed in detail in Chapter 3. Also, all theoretical conclusions are confirmed by experiments with various synthetic and real datasets.

The theoretical results of Chapter 3 help us to formulate main recommendations for tuning of the semi-supervised learning method. We recommend, if possible, to choose labelled points with large degrees. Then, adopt the Standard Laplacian method with α in the upper-middle range of the interval $(0, 1)$. If finding large degree points is not feasible or recall is more important than precision for small classes, choose the PageRank based method. When using the PageRank based method parameter α should be chosen rather close to one but at the same time not too close to avoid numerical instability.

As mentioned in the very beginning of the Introduction, one of our motivating applications is the classification of content and users in P2P systems. Thus, we dedicate a whole Chapter 4 to this particular application and use this application to demonstrate theoretical conclusions obtained in the two preceding chapters. The graph-based semi-supervised learning appears to be a very well suited technique for the classification of content and users in P2P systems. Just to provide flavor of the results, in the dataset of 1 126 670 users, using only 50 labelled points for each language, we are able to classify the users according to their preferred language with more than 95% accuracy. One of the technical advantages of the graph-based semi-supervised learning methods is their ability to scale very well with the data volume.

One of the main conclusions from Chapters 2-4 is that a good choice of labelled data can significantly increase the quality of classification. We have noticed that data points with large (weighted) degree or with large value of PageRank typically represent very well their respective classes. Thus, in the second part of the thesis, in Chapters 5-7, we investigate the question how to find quickly nodes with large centrality measures.

In particular, in Chapter 5 we propose a random walk based method for quick detection of large degree nodes. Since once a random walk comes across a node, we know its degree, the analysis of the algorithm is equivalent to the analysis of hitting events for a Markov chain. We show that on configuration type random graphs our algorithm finds top degree nodes in sublinear time. We design two stopping criteria which allow simple online implementation of the algorithm without any a priori knowledge of the network topology. We also demonstrate that our algorithm works well in practice. For instance, in the UK web graph (symmetrized version) with 18 500 000 nodes, crawled in 2002, our algorithm finds the largest degree node on average three orders of magnitude faster than the Heapsort type algorithm.

In Chapter 6 we suggest and analyse Monte Carlo type methods for quick detection of nodes with large values of Standard and Personalized PageRanks. On one hand, this can be useful for

quick selection of high quality labelled data points and on the other hand, we can apply the method to discover quickly coarse classification or to find higher quality class representatives. In [14] it is shown that Monte Carlo estimation for large PageRank values requires about the same number of operations as one iteration of the power iteration method. In Chapter 6 we show that the Monte Carlo algorithms require an incomparably smaller number of operations when our goal is to detect a top-k list with k not large. In our test on the Wikipedia entity graph with about 2 million nodes typically few thousands of operations are enough to detect the top-10 list with just two or three erroneous elements. Hence, we obtain a relaxation of the top-10 list problem with just about 1-5% of operations required by one power iteration. We would like to emphasize that the Monte Carlo approach allows easy online and parallel implementation. As a by-product, we have also established a new matrix-form central limit theorem for Markov chains.

Finally, in Chapter 7 we propose and analyse a new centrality measure — the alpha current flow centrality. The alpha current flow centrality can be considered as a generalization unifying betweenness centrality (see e.g., [28, 70]) and PageRank. Furthermore, our measure takes into account not only shortest paths as the classical betweenness centrality does [28], but also contribution from all paths, giving more weight to the contribution coming from shorter paths [70]. As recently was pointed out in [63], the betweenness centrality type measure can be used to improve the selection of the labelled data or even as classification function. Furthermore, we demonstrate that the alpha current flow betweenness is a good measure to predict the vulnerability of networks.

In Chapter 9 we conclude with main recommendations for the graph-based semi-supervised learning and the detection of most central / important nodes in a network. We also discuss possible future research directions.

In this introductory chapter we have cited only the main references on the studied topic. At the beginning of each chapter we shall discuss in detail works related to the material of that chapter.

Part I

Graph-based Semi-supervised Learning Methods

2

OPTIMIZATION FRAMEWORK FOR SEMI-SUPERVISED LEARNING METHODS

2.1 Introduction and summary of the results

Semi-supervised classification is a special form of classification. Traditional classifiers use only labeled data to train. Semi-supervised learning methods use large amount of unlabeled data, together with labeled data, to build better classifiers. Semi-supervised learning requires less human effort and gives high accuracy. Graph-based semi-supervised methods use a graph where the nodes are labeled and unlabeled instances in the dataset, and edges (may be weighted) reflect the similarity of instances. These methods usually assume label smoothness over the graph (see the excellent books on the graph-based semi-supervised learning [33, 85]). In this work we often omit "graph-based" term as it is clear that we only consider graph-based semi-supervised learning methods.

Up to the present, most literature on the graph-based semi-supervised learning studied the following two methods: the Standard Laplacian based method (see e.g., [82]) and the Normalized Laplacian based method (see e.g., [81]). Here we propose a generalized optimization framework which implies the above two methods as particular cases. Moreover, our generalized optimization framework gives PageRank based method as another particular case. The PageRank based methods have been proposed in [10] as a classification stage in a clustering method for large hyper-text document collections. In [10] only a linear algebraic formulation was proposed but not the optimization formulation. A great advantage of the PageRank based method is that it has a quasi-linear complexity. We observe that if one takes the method of

[7] for detecting local cuts and takes seeds in [7] as the labelled data and considers sweeps as classification functions, then because the degrees of data points in different sweeps are the same, the resulting method will be equivalent to the semi-supervised method proposed in [10]. In [34] another method based on PageRank has been proposed. However, the method of [34] cannot be scaled to large datasets as it is based on the K-means method.

The generalized optimization framework allows us to provide intuitive interpretation of the differences between particular cases. Using the terminology of random walks on graphs, in Chapter 3 we discuss further differences among the semi-supervised learning methods. The generalized optimization framework has only two free parameters to tune. By choosing the first parameter, we vary the level of credit that we give to nodes with large degree. By choosing the second parameter, the regularization parameter, we choose a trade-off between the closeness of the classification function to the labeling function and the smoothness of the classification function over the graph. We would like to note that the particular cases of our generalized optimization framework (Standard Laplacian method and PageRank based method) are among the best performing semi-supervised methods as reported in the very extensive comparative study [40].

We study sensitivity of the methods with respect to the value of the regularization parameter. We conclude that only the PageRank based method shows robustness with respect to the choice of the value of the regularization parameter.

In this chapter we illustrate our theoretical results and obtain further insights from two datasets. The first dataset is a graph of co-appearance of the characters in the novel Les Miserables. The second data set is a collection of articles from Wikipedia for which we have expert classification. We have compared the quality of classification of the graph-based semi-supervised learning methods with the quality of classification based on Wikipedia categories. It is remarkable to observe that with just few labeled points and only using the hyper-text links, the graph-based semi-supervised methods perform nearly as good as Wikipedia categories in terms of precision and even better in terms of recall. With the help of the two datasets we confirm that the PageRank based method is more robust than the other two methods with respect to the value of the regularization parameter and with respect to the choice of labeled points.

This chapter is mainly based on the article:
K. Avrachenkov, P. Gonçalves, A. Mishenin and M. Sokol, “Generalized Optimization Frameworks for Semi-supervised Learning”, In the SIAM conference on Data Mining, SDM 2012.

2.2 Generalized Optimization Framework

Suppose we need to classify N data points into K classes and assume P data points are labelled. That is, we know the class to which each labelled point belongs. Denote by V_k, the set of labelled points in class $k = 1, ..., K$. Thus, $|V_1| + ... + |V_K| = P$.

The graph-based semi-supervised learning approach uses a weighted graph connecting data points. The weight matrix, or similarity matrix, is denoted by W. Here we assume that W is symmetric and the underlying graph is connected. Each element $w_{i,j}$ represents the degree of similarity between data points i and j. Denote by D a diagonal matrix with its (i,i)-element equal to the sum of the i-th row of matrix W: $d_i = \sum_{j=1}^{N} w_{i,j}$.

In some applications the similarity graph is naturally provided by the application in question (e.g., P2P graphs, Hyper-text graphs, graph of social connections). In other applications, the data points are represented by vectors of attributes (X_i, $i = 1, ..., N$) and the weights of the similarity matrix can be calculated for instance using Radial Basis Function (RBF)

$$w_{ij} = \exp(-\|X_i - X_j\|^2/\gamma),$$

or the adjacency matrix can be constructed using the k-Nearest Neighbors (kNN) method (see e.g., [33]).

Define an $N \times K$ matrix Y as

$$Y_{ik} = \begin{cases} 1, & \text{if } i \in V_k, \text{ i.e., point } i \text{ is labelled as a class } k \text{ point,} \\ 0, & \text{otherwise.} \end{cases}$$

We refer to each column Y_{*k} of matrix Y as a labeling function. Also define an $N \times K$ matrix F and call its columns F_{*k} classification functions. The general idea of the graph-based semi-supervised learning is to find classification functions so that on the one hand they will be close to the corresponding labeling function and on the other hand they will change smoothly over the graph associated with the similarity matrix. This general idea can be expressed with the help of optimization formulation. In particular, there are two widely used optimization frameworks. The first formulation, the Standard Laplacian based formulation [82], is as follows:

$$\min_F \{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\|F_{i*} - F_{j*}\|^2 + \mu \sum_{i=1}^{N} d_i\|F_{i*} - Y_{i*}\|^2\}, \tag{2.1}$$

and the second, the Normalized Laplacian based formulation [81], is as follows:

$$\min_F \{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\|\frac{F_{i*}}{\sqrt{d_i}} - \frac{F_{j*}}{\sqrt{d_j}}\|^2 + \mu \sum_{i=1}^{N} \|F_{i*} - Y_{i*}\|^2\}, \tag{2.2}$$

where μ is a regularization parameter. In fact, the parameter μ represents a trade-off between the closeness of the classification function to the labeling function and its smoothness.

Once the classification functions are obtained, the points are classified according to the rule

$$F_{ik} > F_{ik'}, \forall k' \neq k \quad \Rightarrow \quad \text{Point } i \text{ is classified into class } k = \arg\max_{k'} F_{ik'}.$$

The ties can be broken in arbitrary fashion.

Here we propose a generalized optimization framework, which has as particular cases the two above mentioned formulations. Namely, we suggest the following optimization formulation

$$\min_F \{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\|d_i^{\sigma-1}F_{i*} - d_j^{\sigma-1}F_{j*}\|^2 + \mu\sum_{i=1}^{N} d_i^{2\sigma-1}\|F_{i*} - Y_{i*}\|^2\}. \tag{2.3}$$

In addition to the Standard Laplacian formulation ($\sigma = 1$) and the Normalized Laplacian formulation ($\sigma = 1/2$), we obtain the third very interesting case when $\sigma = 0$. We show below that this particular case corresponds to PageRank based clustering [10], for which (2.3) can be rewritten as:

$$\min_F \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\|\frac{F_{i*}}{d_i} - \frac{F_{j*}}{d_j}\|^2 + \mu\sum_{i=1}^{N}\frac{1}{d_i}\|F_{i*} - Y_{i*}\|^2.$$

Since the objective function of the generalized optimization framework is a sum of a positive semi-definite quadratic form and a positive quadratic form, we can state the following proposition.

Proposition 2.1 *The objective of the generalized optimization framework for semi-supervised learning is a convex function.*

One way to find F is to apply one of many efficient optimization methods for convex optimization. Another way to find F is to find it as a solution of the first order optimality condition. Fortunately, we can even find F in explicit form.

Proposition 2.2 *The classification functions for the generalized semi-supervised learning are given by*

$$F_{*k} = \frac{\mu}{2+\mu}\left(I - \frac{2}{2+\mu}D^{-\sigma}WD^{\sigma-1}\right)^{-1} Y_{*k}, \tag{2.4}$$

for $k = 1, ..., K$.

Proof: The objective function of the generalized semi-supervised learning framework can be rewritten in the following matrix form

$$Q(F) = 2\sum_{k=1}^{K} F_{*k}^T D^{\sigma-1}LD^{\sigma-1}F_{*k}$$
$$+\mu\sum_{k=1}^{K}(F_{*k} - Y_{*k})^T D^{2\sigma-1}(F_{*k} - Y_{*k}),$$

where $L = D - W$ is the Standard Laplacian. The first order optimality condition $D_{F_{*k}} Q(F) = 0$ gives

$$2F_{*k}^T(D^{\sigma-1}LD^{\sigma-1} + D^{\sigma-1}L^TD^{\sigma-1}) \\ +2\mu(F_{*k} - Y_{*k})^TD^{2\sigma-1} = 0.$$

Multiplying the above expression from the right hand side by $D^{-2\sigma+1}$, we obtain

$$2F_{*k}^T(D^{\sigma-1}(L + L^T)D^{-\sigma}) + 2\mu(F_{*k} - Y_{*k})^T = 0.$$

Then, substituting $L = D - W$ and rearranging the terms yields

$$F_{*k}^T(2I - D^{\sigma-1}(W + W^T)D^{-\sigma} + \mu I) - \mu Y_{*k}^T = 0.$$

Since W is a symmetric matrix, we obtain

$$F_{*k}^T(2I - 2D^{\sigma-1}WD^{-\sigma} + \mu I) - \mu Y_{*k}^T = 0.$$

Using the fact that $2I - 2D^{\sigma-1}WD^{-\sigma} + \mu I$ is invertible matrix, we have

$$F_{*k}^T = \mu Y_{*k}^T(2I - 2D^{\sigma-1}WD^{-\sigma} + \mu I)^{-1},$$

which proves the proposition.

As a corollary, we have explicit expressions for the classification functions for the three mentioned above particular semi-supervised learning methods. Namely, from expression (2.4) we derive

- if $\sigma = 1$, the Standard Laplacian method [82]:

$$F_{*k} = \frac{\mu}{2+\mu}(I - \frac{2}{2+\mu}D^{-1}W)^{-1}Y_{*k},$$

- if $\sigma = 1/2$, the Normalized Laplacian method [81]:

$$F_{*k} = \frac{\mu}{2+\mu}(I - \frac{2}{2+\mu}D^{-\frac{1}{2}}WD^{-\frac{1}{2}})^{-1}Y_{*k},$$

- if $\sigma = 0$, the PageRank based method [10]:

$$F_{*k} = \frac{\mu}{2+\mu}(I - \frac{2}{2+\mu}WD^{-1})^{-1}Y_{*k}.$$

Let us now explain why the case $\sigma = 0$ corresponds to the PageRank based clustering method. Denote $\alpha = 2/(2+\mu)$ and write F_{*k} in a transposed form

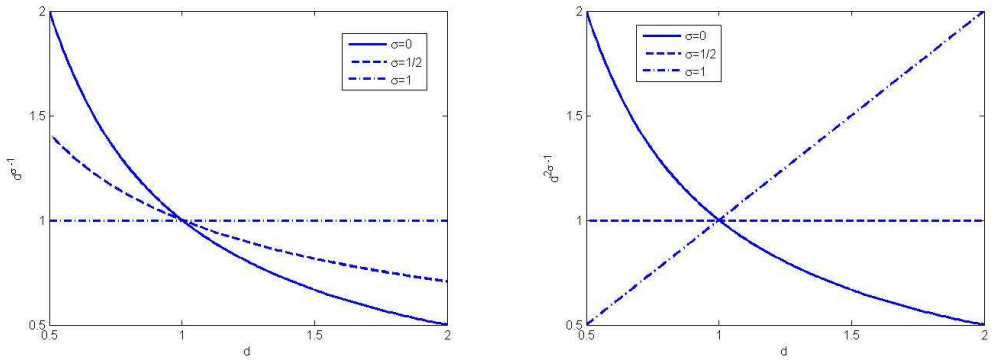
$$F_{*k}^T = (1-\alpha)Y_{*k}^T(I - \alpha D^{-1}W)^{-1}.$$

If the labeling functions are normalized, this is exactly an explicit expression for PageRank [66, 54]. This expression was used in [10] but no optimization framework was provided.

Note that $D^{-1}W$ represents the transition probability matrix for the random walk on the similarity graph. Then, the (i,j)-th element of the matrix $(I - \alpha D^{-1}W)^{-1}$ gives the expected number of visits to node j starting from node i until the random walk restarts with probability $1-\alpha$. This observation provides the following probabilistic interpretation for the Standard Laplacian and PageRank based methods. In the Standard Laplacian method, F_{ik} gives up to a multiplicative constant the expected number of visits before restart to the labeled nodes of class k if the random walk starts from node i. In the PageRank based method with normalized labeling functions, F_{ik} gives up to a multiplicative constant the expected number of visits to node i, if the random walk starts from a uniform distribution over the labeled nodes of class k.

The random walk approach can explain why in some cases Standard Laplacian and PageRank based methods provide different classifications. For instance, consider a case when a node v is directly connected to the labeled nodes k_1 and k_2 belonging to different classes. Furthermore, let the labeled node k_1 have a higher degree than the node k_2 and let the node k_1 belong to a denser cluster than node k_2. From [13] we know that the expected number of visits to node j starting from node i until the restart is equal to the product of the probability to visit node j before the absorption and the expected number of returns to node j starting from node j. Then, the PageRank based method will classify the node v into the class of the labeled node k_2 as it is more likely that the random walk misses the node v starting from node k_1. In other words, when the random walk starts from k_2, there are less options how to choose a next node and it is more likely to choose node v as a next node. In the Standard Laplacian method we need to compare the average number of visits to the labeled nodes starting from the node v. Since the random walk can reach either node k_1 or node k_2 in one step the probabilities of hitting these nodes before absorption are similar and what matters is how dense are the classes. If the class associated with the labeled node k_1 is more dense than the class associated with the labeled node k_2, the node v will be classified to the class associated with k_1. This reasoning will be made rigorous in the next chapter.

Based on the formulation (2.3), we could give some further intuitive interpretation for various cases of the generalized semi-supervised learning. Let us consider the first term in the r.h.s. sum of (2.3), which corresponds to the smoothness component. Figure 2.1(a) shows that if $\sigma < 1$ we do not give much credit to the connections between points with large degrees. Let us now consider the second term which corresponds to the fitting function. Figure 2.1(b) shows that $\sigma < 1/2$ does not give much credit to samples that pertain to a dense cluster of points (i.e. d_{ii} is large), whereas samples that are relatively isolated in the feature space (corresponding to small value of d_{ii}), are given higher confidence. If $\sigma = 1$, the node degree does not have any influence. And if $\sigma > 1/2$, we consider that the nodes with higher weighted degree are more



(a) Smoothness term. (b) Fitting term.

Figure 2.1: Fitting and smoothness terms.

important than the nodes with smaller degree.

Next we analyze the limiting behavior of the semi-supervised learning methods when $\mu \to 0$ ($\alpha \to 1$). Towards this goal, we shall use the Blackwell series expansion [22, 74]

$$(1-\alpha)\left(I - \alpha D^{-1}W\right)^{-1} = \underline{1}\pi + (1-\alpha)H + o(1-\alpha), \tag{2.5}$$

where π is the stationary distribution of the standard random walk ($\pi D^{-1}W = \pi$), $\underline{1}$ is a vector of ones of appropriate dimension and $H = (I - D^{-1}W + \underline{1}\pi)^{-1} - \underline{1}\pi$ is the deviation matrix. We note that since the similarity matrix W is symmetric, the random walk governed by the transition matrix $D^{-1}W$ is time-reversible and its stationary distribution is given in the explicit form

$$\pi = (\underline{1}^T D \underline{1})^{-1} \underline{1}^T D. \tag{2.6}$$

We transform the expression (2.4) to a more convenient form.

$$\begin{aligned} F_{*k} &= \frac{\mu}{2+\mu}\left(I - \frac{2}{2+\mu}D^{-\sigma}WD^{\sigma-1}\right)^{-1} Y_{*k} \\ &= \frac{\mu}{2+\mu}\left(D^{-\sigma}\left(I - \frac{2}{2+\mu}WD^{-1}\right)D^{\sigma}\right)^{-1} Y_{*k} \\ &= \frac{\mu}{2+\mu}D^{-\sigma}\left(I - \frac{2}{2+\mu}WD^{-1}\right)^{-1} D^{\sigma}Y_{*k}. \end{aligned}$$

Transposing and using the fact that W is symmetric, we obtain

$$F_{*k}^T = (1-\alpha)Y_{*k}^T D^{\sigma}\left(I - \alpha D^{-1}W\right)^{-1} D^{-\sigma}. \tag{2.7}$$

Combining (2.7), (2.5) and (2.6), we can write

$$F_{*k}^T = (\underline{1}^T D \underline{1})^{-1} Y_{*k}^T D^{\sigma} \underline{1}\underline{1}^T D^{1-\sigma} + (1-\alpha)Y_{*k}^T D^{\sigma} H D^{-\sigma} + o(1-\alpha).$$

In particular, we have

$$F_{ik} = \frac{d_i^{1-\sigma}}{\sum_{j=1}^{N} d_j} \sum_{p \in V_k} d_p^{\sigma} + (1-\alpha) d_i^{-\sigma} \sum_{p \in V_k} d_p^{\sigma} H_{pi} + o(1-\alpha), \tag{2.8}$$

and, consequently, if $\sum_{p \in V_k} d_p^{\sigma} \neq \sum_{p \in V_k'} d_p^{\sigma}$ for some k and k', in the case when the parameter α is close to 1 (equivalently when μ is close to 0), then all points will be classified into the classes with the largest value of $\sum_{p \in V_k} d_p^{\sigma}$. An interesting exception is the case when $\sigma = 0$ and $|V_k| = const(k)$. In such a case, the zero order terms in the Blackwell expansions for the classification functions are the same for all classes and we need to compare the first order terms. Recall [52] that there is a connection between the mean first passage time of the standard random walk from node i to node j, m_{ij}, and the elements of the deviation matrix, namely, $m_{ij} = (\delta_{ij} + H_{jj} - H_{ij})/\pi_j$, where δ_{ij} is the Kronecker delta. If $\sigma = 0$ and $|V_k| = const(k)$, substituting (2.8) into $F_{ik} - F_{ik'} > 0$ with $H_{pi} = H_{ii} - \pi_i m_{pi}$ for $i \neq p$ results in the condition

$$\sum_{s \in V_{k'}} m_{si} > \sum_{p \in V_k} m_{pi}.$$

This condition has a clear probabilistic interpretation: point i is classified into class k if the sum of mean passage times from the labelled points to point i is smallest for class k over all classes.

The conclusion is that the PageRank based method is more robust to the choice of the regularization parameter than the other graph-based semi-supervised learning methods.

Illustrating example: to illustrate the limiting behaviour of the methods we generated an artificial example of the planted partition random graph model [35] with two classes with 100 nodes in each class. The probability of link creation inside the first class is 0.3 and the probability of link creation inside the second class is 0.1. So the first class is three times denser than the second class. The probability of link creation between two classes is 0.05. We have generated a sample of this random graph model. In each class we have chosen just one labelled point. In the first class we have chosen as the labelled point the point with the smallest degree (degree=28, 24 edges inside the class and 4 edges leading outside). In the second class we have chosen as the labelled point the point with the largest degree (degree=31, 27 edges inside the class and 4 edges leading outside). We have indeed observed that the second class attracts all points when α is close to one for all semi-supervised methods except for the PageRank based method. This is in accordance with theoretical conclusions as the labelled point in the second class has a larger weight than the labelled point in the first class. It is interesting to observe that in this example the first class looses all points when α is close to one even though the first class is denser then the second one.

2.3 Experiments

In this section we apply the developed theory to two datasets. The first dataset is the network of interactions between major characters in the novel Les Miserables. If two characters participate in one or more scenes, there is a link between these two characters. The second dataset is a subset of Wikipedia pages. Wikipedia articles correspond to the data points and hyper-text links correspond to the edges of the similarity graph. We disregard the direction of the hyper-text links.

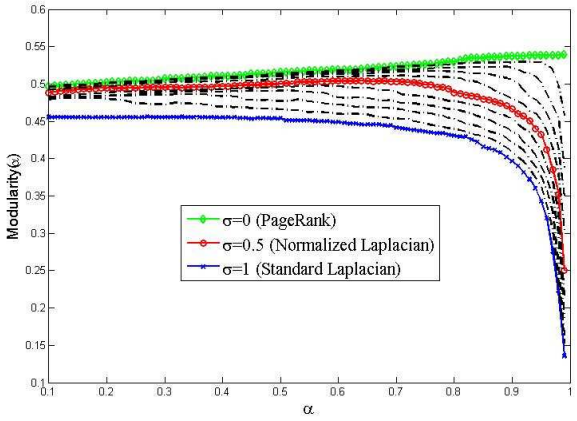
2.3.1 Les Miserables example

The graph of the interactions of Les Miserables characters has been compiled by Knuth [53]. There are 77 nodes and 508 edges in the graph. Using the betweenness based algorithm of Newman [71] we obtain 6 clusters which can be identified with the main characters: Valjean (17), Myriel (10), Gavroche (18), Cosette (10), Thenardier (12), Fantine (10), where in brackets we give the number of nodes in the respective cluster. We have generated randomly 100 times labeled points (one labeled point per cluster). In Figure 2.2(a) we plot the modularity measure averaged over 100 experiments as a function of α for methods with different values of σ ranging from 0 to 1 with granularity 0.1. The modularity measure is based on the inter-cluster link density and the average link density and reflects the quality of clustering [71]. From Figure 2.2(a) we conclude that on average the PageRank based method performs best in terms of modularity and it is robust with respect to the choice of the regularization parameter. In particular, we observe that as was predicted by the theory the Standard Laplacian method and Normalized Laplacian method perform badly when α is close to 1 (one class attracts all instances). The PageRank based method is robust even for the values of α which are very close to one. We return to this example in the next chapter, discussing the interpretations based on random walks.

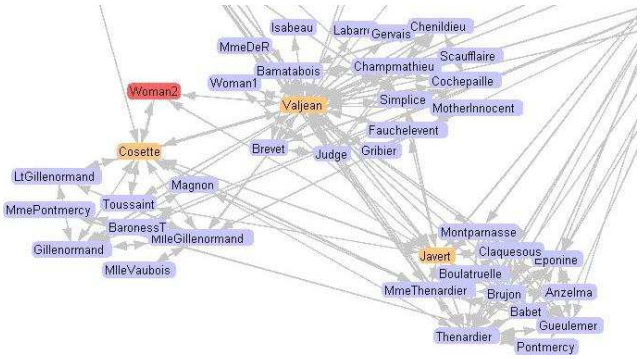
2.3.2 Wikipedia-math example

The second dataset is derived from the English language Wikipedia. In this case, the similarity graph is constructed by a slight modification of the hyper-text graph. Each Wikipedia article typically contains links to other Wikipedia articles which are used to explain specific terms and concepts. Thus, Wikipedia forms a graph whose nodes represent articles and whose edges represent hyper-text inter-article links. For our experiments we took a snapshot (dump) of Wikipedia from January 30, 2010[1]. Based on this dump we have extracted outgoing links for other articles. The links to special pages (categories, portals, etc.) have been ignored. In the

[1] `http://download.wikimedia.org/enwiki/20100130`



(a) Modularity as a function of α.



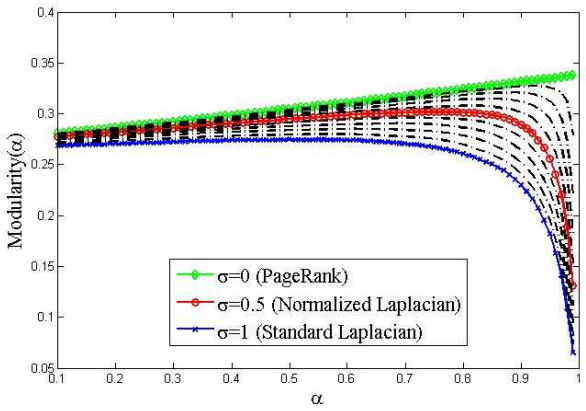
(b) Difference in classifications.

Figure 2.2: Les Miserables example.

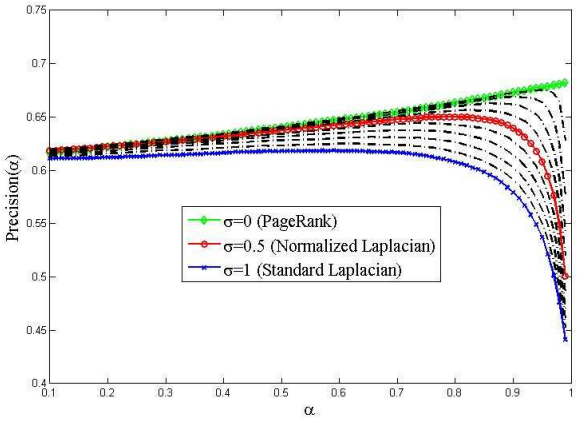
present experiment we did not use the information about the direction of links, so the graph in our experiments is undirected. Then we have built a subgraph with mathematics related articles, a list of which was obtained from "List of mathematics articles" page from the same dump. In the present experiments we have chosen the following three mathematical topics: "Discrete mathematics" (DM), "Mathematical analysis" (MA), "Applied mathematics" (AM). With the help of AMS MSC Classification [2] and experts we have classified related Wikipedia mathematical articles into the three above mentioned topics. According to the expert annotation we have built a subgraph of the Wikipedia mathematical articles providing imbalanced classes DM (106), MA (368) and AM (435). The subgraph induced by these three topics is connected and contains 909 articles. Then, the similarity matrix W is just the adjacency matrix of this subgraph. Thus, $w_{ij} = 1$ means that Wikipedia article i is connected with Wikipedia article j. Then, we have cho-

[2] `http://www.ams.org/mathscinet/msc/msc2010.html`

sen uniformly at random 100 times 5 labeled nodes for each class. In Figure 2.3(a) we plot the modularity averaged over 100 experiments as a function of α for methods with different values of σ ranging from 0 to 1 with granularity 0.1. Figure 2.3(a) confirms the observations obtained from Les Miserable dataset that the PageRank based method ($\sigma = 0$) has the best performance in terms of the modularity measure. Next, in Figure 2.3(b) we plot the precision as a function of the regularization parameter for each of the three methods with respect to the expert classification. For the most values of α the PageRank based method performs better than all the other methods and shows robust behaviour when the regularization parameter approaches one. This is in agreement with the theoretical conclusions at the end of Section 2. Both Figure 2.3(a) and Figure 2.3(b) demonstrate that the PageRank based method is also more robust than the other two methods with respect to the choice of labeled points.



(a) Modularity as a function of α.



(b) Precision as a function of α.

Figure 2.3: Wikipedia-math example: Modularity and Precision of the classification.

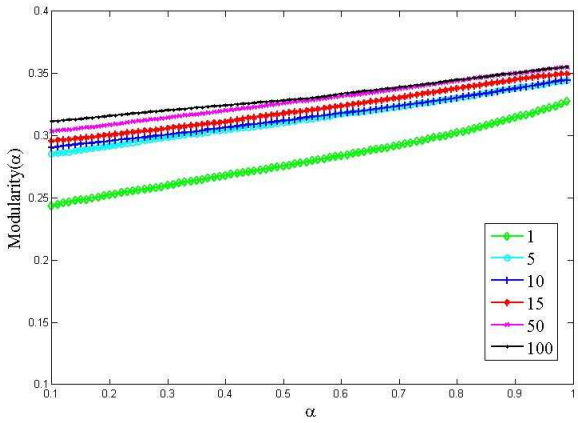
Figure 2.3(a) and Figure 2.3(b) also suggest that we can use the modularity measure as a good criterion for the choice of the regularization parameter for the Standard Laplacian and Normalized Laplacian methods. Now, let us investigate the effect of the quantity of the labelled data on the quality of the classification. Figures 2.4(a) 2.4(b) and 2.4(c) show that on average the modularity of the classification increases when we increase the quantity of the labelled data. Moreover, the quality of classification improves significantly when we increase the quantity of labelled data for each class from few points to about 50 points. The further increase of the quantity of the labelled data does not result in significant improvement in classification quality. The same behaviour manifests itself with respect to the precision measure (see Figures 2.5(a) 2.5(b) and 2.5(c)).

Both Les Miserables and Wikipedia-math datasets indicate that for the PageRank based method it is better to choose the value of the regularization parameter as close to one as possible but at the same time keeping the system numerically stable and efficient. This is an example of the singular perturbation phenomena [11, 80]

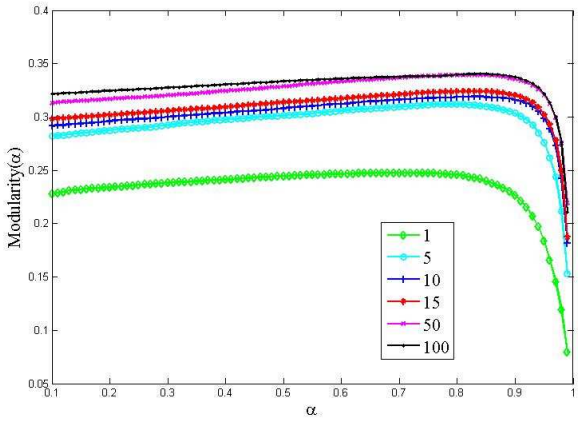
We have also compared the results obtained by the semi-supervised learning methods with the classification provided by Wikipedia Categories. As Wikipedia categories we have chosen: `Applied_mathematics`, `Mathematical_analysis` and `Discrete_mathematics`. It turns out that the precision of the Wikipedia categories with respect to the expert classification is 78% (with 5 random labelled points the PageRank based method can achieve about 68%). However, the recall of the Wikipedia categorization is 72%. With the help of the semi-supervised learning approach we have classified all articles. It is quite interesting to observe that just using the link information the semi-supervised learning can achieve precision nearly as good as the Wikipedia categorization produced by hard work of many experts and the semi-supervised learning can do even better in terms of recall.

2.4 Conclusions

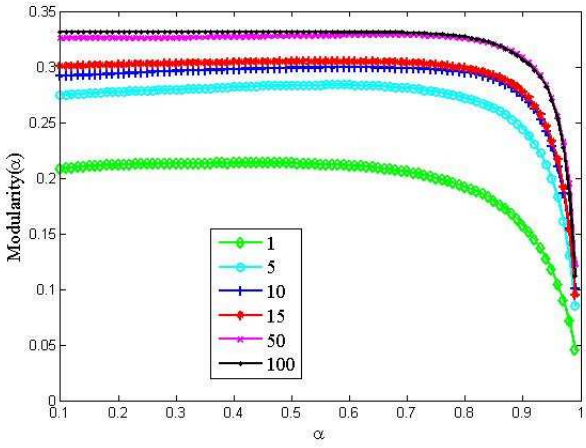
We have developed a generalized optimization approach for the graph-based semi-supervised learning which implies as particular cases the Standard Laplacian, Normalized Laplacian and PageRank based methods and provides the new ones based on parameter σ. We have also characterized the limiting behaviour of the methods as $\alpha \to 1$ which based on the weight of the labelled points. We have illustrated theoretical results with the help of Les Miserables example and Wikipedia-math example. Both theoretical and experimental results demonstrate that the PageRank based method outperforms the other methods in terms of clustering modularity and robustness when the labelled points are chosen randomly. We propose to use the modularity measure for the choice of the regularization parameter in the cases of the Standard Laplacian method and the Normalized Laplacian method. In the case of the Pagerank based method we suggest to choose the value of the regularization parameter as close to one as possible but at the same time keeping the system numerically stable and efficient. It appears that remarkably we can classify the Wikipedia articles with very good precision and perfect recall employing only the information about the hyper-text links.



(a) Modularity as a function of α for PageRank.

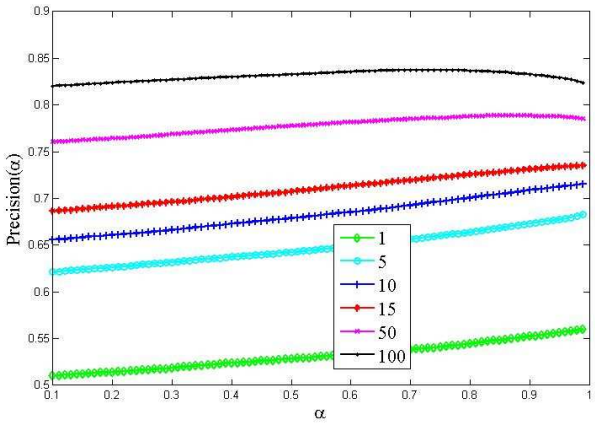


(b) Modularity as a function of α for Normalized Laplacian.

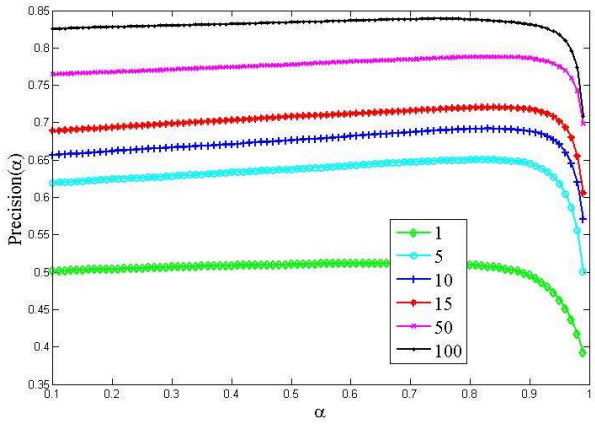


(c) Modularity as a function of α for Standard Laplacian.

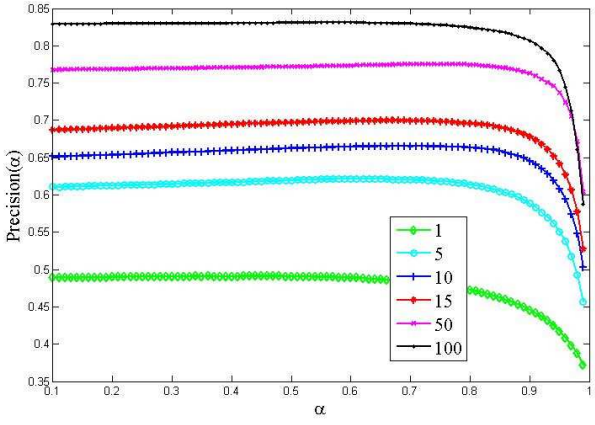
Figure 2.4: Wikipedia-math example: Modularity of the classification for different number of labels.



(a) Precision as a function of α for PageRank.



(b) Precision as a function of α for Normalized Laplacian.



(c) Precision as a function of α for Standard Laplacian.

Figure 2.5: Wikipedia-math example: Precision of the classification for different number of labels.

3

RANDOM WALK APPROACH FOR SEMI-SUPERVISED LEARNING METHODS

3.1 Introduction and summary of the results

In Chapter 2 we have proposed a generalized optimization formulation which gives as important particular cases: the Standard Laplacian method, the Normalized Laplacian method and the PageRank based method. In the current chapter we provide more insights about the differences among the semi-supervised learning methods based on random walk theory, and give recommendations on how to choose the kernel and labelled points (of course, when there is some freedom in the choice of labelled points). In particular, we try to answer the questions: which kernel (or which values of σ and μ) one needs to choose? and which points to label if we have some freedom with respect to labelling points? It turns out that these questions are not independent and one has to choose the kernel depending on the information available while labelling the points. We show that if the labelled points are chosen uniformly at random, the PageRank based method is the best choice for the semi-supervised kernel. On the other hand, if one can choose labelled points with large degrees or we know that labelled points given to us have large degrees, the Standard Laplacian method is the best choice. We illustrate the theoretical conclusions on an analytically tractable characteristic network example, on clustered preferential attachment model and with applications to handwritten digits classification and to Les Miserables social network example. In particular, our theoretical results explain why surprisingly all the semi-supervised methods produce very similar classifications on the dataset of handwritten digits. We conclude the chapter with general recommendations on the choice of

the semi-supervised learning method and labelled points.

In this chapter we extensively use the theory of random walks on graphs. For the background on the random walks on graphs we recommend the excellent survey [60] and the online book [5].

This chapter is mainly based on the article:
K. Avrachenkov, P. Gonçalves and M. Sokol, "On the Choice of Kernel and Labelled Data in Semi-supervised Learning Methods", In the 10th International Workshop on Algorithms and Models for the Web Graph, WAW 2013.

3.2 General theoretical considerations

Let us use the theory of random walk on graph to understand the semi-supervised learning methods. In particular, it is very helpful to consider a random walk with absorption $\{S_t \in \{1,...,N\},\ t = 0,1,...\}$. At each step with probability α the random walk chooses next node among its neighbours uniformly and with probability $1-\alpha$ goes into the absorbing state. The probabilities of visiting nodes before absorption given the random walk starts at node j, $S_0 = j$, are provided by the distribution

$$\mathrm{ppr}(j) = (1-\alpha)e_j^T \left(I - \alpha D^{-1}W\right)^{-1}, \tag{3.1}$$

which is the personalized PageRank vector with respect to seed node j [45]. Here e_j denotes the j-th element of the standard basis.

Now we are ready to formulate the first result explaining the classification by the semi-supervised learning methods.

Theorem 3.1 *Data point i is classified by the generalized semi-supervised learning method (2.3) into class k, if*

$$\sum_{p\in V_k} d_p^\sigma q_{pi} > \sum_{s\in V_{k'}} d_s^\sigma q_{si}, \quad \forall k' \neq k, \tag{3.2}$$

where q_{pi} is the probability of reaching state i before absorption if $S_0 = p$.

Proof: Since $Y_{*k}^T = \sum_{p\in V_k} e_p^T$ and $F_{ik} = F_{*k}^T e_i$, from (2.7) we obtain

$$F_{ik} = \sum_{p\in V_k} d_p^\sigma (1-\alpha) e_p^T \left(I - \alpha D^{-1}W\right)^{-1} e_i d_i^{-\sigma} = \frac{1}{d_i^\sigma} \sum_{p\in V_k} d_p^\sigma \mathrm{ppr}_i(p). \tag{3.3}$$

It has been shown in [13] that

$$\left(I - \alpha D^{-1}W\right)^{-1}_{pi} = q_{pi}\left(I - \alpha D^{-1}W\right)^{-1}_{ii},$$

where $(\cdot)^{-1}_{pi}$ denotes the (p,i)-th element of the inverse matrix. Multiplying the above equation by $(1-\alpha)$ yields

$$\mathrm{ppr}_i(p) = q_{pi}\mathrm{ppr}_i(i). \tag{3.4}$$

Thus, using relation (3.4) and equation (3.3), we conclude that for point i to be classified into class k we need

$$F_{ik} - F_{ik'} = \frac{\mathrm{ppr}_i(i)}{d_i^\sigma}\left(\sum_{p\in V_k} d_p^\sigma q_{pi} - \sum_{s\in V_{k'}} d_s^\sigma q_{si}\right) > 0, \quad \forall k' \neq k,$$

or, equivalently (3.2). ∎

Let us discuss the implications of Theorem 3.1. First, it is very interesting to observe that, using (3.2), one can decouple the effects from the choice of α and σ. A change in the value of α only influences the factor q_{pi} and a change in the value of σ only affects the factor d_p^σ. Second, the results of Theorem 3.1 are consistent with the conclusions obtained with the help of the Blackwell expansion. When α goes to one, q_{pi} goes to one and indeed classes with the largest value of $\sum_{p\in V_k} d_p^\sigma$ attract all points. Thus, the case of $\sigma = 0$ and $|V_k| = const(k)$ is especially interesting. In this case there is stability of classification even when α is close to one. Third, if $\sigma = 0$ and $|V_k| = const(k)$, one can expect that smaller classes will attract a larger number of "border points" than larger classes. Suppose that class k is smaller than class k'. Then, it is natural to expect that $q_{pi} > q_{si}$ with $p \in V_k$ and $s \in V_{k'}$. This observation will be confirmed by examples in the next section. This effect, if needed, can be compensated by increasing σ away from zero. And finally, fourth, we have the following rather surprising conclusion.

Corollary 3.1 *If labelled points have the same degree* ($d_p = d$, $p \in V_k$, $k = 1, ..., K$)*, all considered semi-supervised learning methods provide the same classification.*

Now with the help of the following lemma, we can obtain another alternative condition for semi-supervised learning classification.

Lemma 3.1 *If the graph is undirected* ($W^T = W$)*, then the following relation holds*

$$ppr_j(i) = \frac{d_j}{d_i} ppr_i(j). \tag{3.5}$$

Proof: We can rewrite (3.1) as follows

$$ppr(i) = (1-\alpha)e_i^T[D - \alpha W]^{-1}D,$$

and hence,

$$ppr(i)D^{-1} = (1-\alpha)e_i^T[D - \alpha W]^{-1}.$$

Since matrix W is symmetric, $[D - \alpha W]^{-1}$ is also symmetric and we have

$$[ppr(i)D^{-1}]_j = (1-\alpha)e_i^T[D - \alpha W]^{-1}e_j = (1-\alpha)e_j^T[D - \alpha W]^{-1}e_i = [ppr(j)D^{-1}]_i.$$

Thus, $ppr_j(i)/d_j = ppr_i(j)/d_i$, which completes the proof. ∎

Theorem 3.2 *Data point* i *is classified by the generalized semi-supervised learning method* (2.3) *into class* k*, if*

$$\sum_{p\in V_k} \frac{ppr_p(i)}{d_p^{1-\sigma}} > \sum_{s\in V_{k'}} \frac{ppr_s(i)}{d_s^{1-\sigma}}, \quad \forall k' \neq k. \tag{3.6}$$

Proof: Follows from equation (3.3) and Lemma 3.1. ∎

We note that in the statement of Theorem 3.2 the "reversed" PageRank is used instead of the PageRank in (3.3). In particular, this provides another interesting interpretation of the PageRank based method. If we set $\sigma = 0$ in (3.6), it appears that we need to compare the reversed PageRanks divided by the degrees of the labelled points. As already mentioned in the Introduction, if one considers the sweeps from [7] as classification functions, then the degrees of the nodes to be classified are cancelled in the sweeps. However, if we now view the PageRank method in terms of the reversed PageRank, the division by the degree of the PageRank values remains essential. This provides another interesting interpretation of sweeps defined in [7].

3.3 Evaluation

Let us illustrate the theoretical results with the help of a characteristic network example, clustered preferential attachment graph and application to Les Miserables social network example and handwritten digits classification.

Characteristic network example: Let us first consider an analytically tractable network example. Despite its simplicity, it clearly demonstrates major properties of graph-based semi-supervised learning methods. There are two classes, A and B with $|A| = N_1$ and $|B| = N_2$. Each class is represented by a star network. The two classes are connected by a link connecting two leaves. The graph of the model is given in Figure 3.2(a).

The central nodes with indices 1 and $N_1 + N_2$ are the obvious choice for labelled points. In order to determine the classification functions analytically, we need to calculate the matrix $Z = [I - \alpha D^{-1}W]^{-1}$. It is easier to calculate the symmetric matrix $C = [D - \alpha W]^{-1}$. Once the matrix C is calculated, we can immediately retrieve the elements of matrix Z by the formula

$$Z_{ij} = C_{ij}d_j. \tag{3.7}$$

Thus we need to solve a system of equations $[D - \alpha W]C_{*,j} = e_j$. Since we have chosen the central nodes as labelled points and due to the symmetry of the graph, we actually need to solve only one system for $j = 1$ of six equations

$$\begin{aligned}
(N_1 - 1)C_{1,1} - (N_1 - 2)\alpha C_{2,1} - \alpha C_{N_1,1} &= 1 \\
C_{2,1} &= \alpha C_{1,1} \\
C_{N_1-1,1} &= \alpha C_{1,1} \\
-\alpha C_{1,1} + 2C_{N_1,1} - \alpha C_{N_1+1,1} &= 0 \\
-\alpha C_{N_1,1} + 2C_{N_1+1,1} - \alpha C_{N_1+N_2,1} &= 0 \\
C_{N_1+2,1} &= \alpha C_{N_1+N_2,1} \\
-\alpha C_{N_1+1,1} - (N_2 - 2)\alpha C_{N_1+2,1} + (N_2 - 1)C_{N_1+N_2,1} &= 0
\end{aligned}$$

Solving the above system, in particular, we obtain

$$C_{N_1,1} = \frac{\alpha(2N_2 - 2 - \alpha^2(2N_2 - 3))}{R}, \tag{3.8}$$

$$C_{N_1+1,1} = \frac{\alpha^2(N_2 - 1 - \alpha^2(N_2 - 2))}{R}, \tag{3.9}$$

with

$$R = (1 - \alpha^2)(-2\alpha^4 N_2 - 2\alpha^4 N_1 + 4\alpha^4 + \alpha^4 N_2 N_1 - 9\alpha^2$$

$$+7\alpha^2 N_2 + 7\alpha^2 N_1 - 5N_2\alpha^2 N_1 + 4N_2 N_1 + 4 - 4N_1 - 4N_2).$$

Consider first the PageRank based method ($\sigma = 0$). According to the theoretical consideration, it is very likely that some points will be misclassified into a smaller class. Suppose that $N_1 < N_2$ and consider border points. The point $N_1 + 1$ will be classified into class B by the PageRank based method if and only if

$$\frac{Z_{1,N_1+1}}{Z_{N_1+N_2,N_1+1}} = \frac{C_{1,N_1+1}}{C_{N_1+N_2,N_1+1}} < 1.$$

Using slightly more convenient notation $n_i = N_i - 1, i = 1, 2$, we can rewrite the above condition as follows:

$$\frac{\alpha(n_2 - \alpha^2(n_2 - 1))}{2n_1 - \alpha^2(2n_1 - 1)} < 1,$$

or, equivalently, $(1-n_2)\alpha^2 + (2n_1 - n_2)\alpha + 2n_1 > 0$. If $2n_1 + 1 > n_2$, the above inequality holds for any $\alpha \in (0, 1)$. And consequently, for any $\alpha \in (0, 1)$ the point $N_1 + 1$ is classified into class B. However, if $2n_1 + 1 < n_2$ (class A is significantly smaller than class B), for $\alpha \in (\bar{\alpha}, 1)$ point $N_1 + 1$ will be erroneously classified into class A. The expression for $\bar{\alpha}$ is given by

$$\bar{\alpha} = \frac{-(n_2 - 2n_1) + \sqrt{(2n_1 + n_2)^2 - 8n_1}}{2(n_2 - 1)}.$$

If we fix the value of n_1 and let n_2 go to infinity, we get $\bar{\alpha} \to 0$. Thus, if the sizes of A and B are very different, the point $N_1 + 1$ will be misclassified for nearly all values of the parameter α.

Now we analyse the performance of the Standard Laplacian method ($\sigma = 1$). According to the general theoretical considerations, the Standard Laplacian method has a tendency to classify more points into a larger class. We consider the classification of the point with index N_1 (still assuming $N_1 < N_2$). It will be classified correctly if and only if

$$\frac{Z_{N_1,1}}{Z_{N_1,N_1+N_2}} > 1,$$

or, equivalently,

$$\frac{n_1(2n_2 - \alpha^2(2n_2 - 1))}{n_2\alpha(n_1 - \alpha^2(n_1 - 1))} > 1$$

which results in the following cubic inequality

$$\alpha^3 n_2(n_1 - 1) - \alpha^2 n_1(2n_2 - 1) - \alpha n_2 n_1 + 2n_2 n_1 > 0.$$

Consider a linear scaling $n_2 = Kn_1, K > 1$. Then, the above inequality can be rewritten in the form

$$\alpha^3\left(1 - \frac{1}{n_1}\right) - \alpha^2\left(2 - \frac{1}{Kn_1}\right) - \alpha + 2 > 0.$$

This inequality can be regarded as a regularly perturbed inequality with respect to $1/n_1$ (see e.g., [11]). If we let n_1 go to infinity, the limiting inequality can be easily factored, i.e.,$(1 -$

$\alpha)(1+\alpha)(2-\alpha) > 0$. Since the perturbation is regular, when n_1 varies in the vicinity of infinity the roots change slightly. In particular, using the implicit function theorem, we can find that the root near 1 changes as follows:

$$\bar{\bar{\alpha}} = 1 - \frac{K-1}{2K}\frac{1}{n_1} + o\left(\frac{1}{n_1}\right).$$

In particular, this means that if the sizes of classes are large, the Standard Laplacian method performs well for nearly all values of α from the interval $(0,1)$. This is in contrast with the PageRank based method.

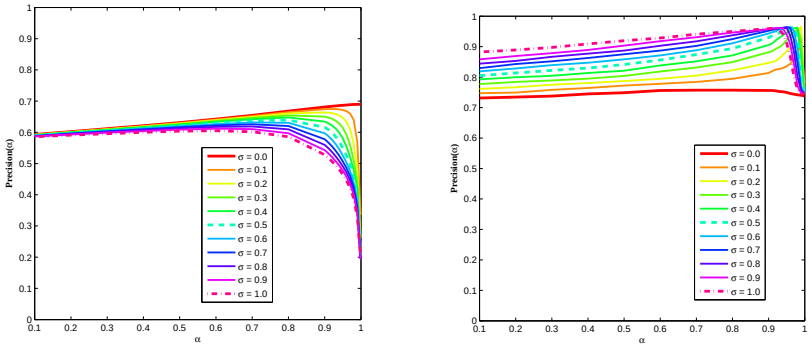
We summarize and illustrate various considered cases by means of numerical examples presented in Table 3.1. Our main conclusion from this characteristic network model is that the PageRank based method is a safe choice as it can misclassify at most one point in this particular example whereas with α close to one the Standard Laplacian method can classify all points in the largest class. On the other hand if parameter α is chosen appropriately, the Standard Laplacian method gives a perfect classification for nearly all values of α, even when classes have many points and very different sizes.

N_1	N_2	PR	SL
20	100	$v_{N_1+1} \mapsto$ A if $\alpha \geq \bar{\alpha} = 0.3849$	$v_{N_1} \mapsto$ B if $\alpha \geq \bar{\bar{\alpha}} = 0.9803$, A $\mapsto$ B if $\alpha \geq 0.9931$
20	200	$v_{N_1+1} \mapsto$ A if $\alpha \geq \bar{\alpha} = 0.1911$	$v_{N_1} \mapsto$ B if $\alpha \geq \bar{\bar{\alpha}} = 0.9780$, A $\mapsto$ B if $\alpha \geq 0.9923$
200	2000	$v_{N_1+1} \mapsto$ A if $\alpha \geq \bar{\alpha} = 0.1991$	$v_{N_1} \mapsto$ B if $\alpha \geq \bar{\bar{\alpha}} = 0.9978$, A $\mapsto$ B if $\alpha \geq 0.9992$

Table 3.1: Comparison between different methods in terms of classification errors

Clustered Preferential Attachment model: Let us now consider a synthetic graph generated according to the clustered preferential attachment model. Our model has 5 unbalanced classes (1500 / 240 / 120 / 100 / 50). Once a node is generated, it has two links which it attaches independently with probability 0.98 within its class and with probability 0.02 outside its class. In both cases a link is attached to a node with probability proportional to the number of existing links. First, we test the case of random labelled points. Five labelled points were chosen randomly for each class and results are averaged over 100 realizations. The precision of classification for various values of σ and α is given in Figure 3.1(a). Then, in each class we have chosen 5 labelled points with maximal degrees. The results of classification are given in Figure 3.1(b). We obtain conclusions consistent with the characteristic network model. If no information is available for assignment of the labelled points, the PageRank method is a safe choice. If one can choose labelled points with large degrees, it is better to use the Standard Laplacian method. There could be a significant gain in precision (roughly from 70% to 95%).

It can be observed that the Standard Laplacian method is not too sensitive to the value of α if we stay well away from $\alpha = 1$.



(a) Random Labelled Points. (b) Max Degree Labelled Points.

Figure 3.1: Clustered Preferential Attachment Model: Precision of the classification.

Application to Les Miserables example: Let us use the random walk based interpretation to explain differences between the Standard Laplacian based method and the PageRank based method in Les Miserables example. Let us consider the node Woman 2 (see Figure 2.2(b)). The node Woman 2 is connected with three other nodes: Valjean, Cosette and Javert. Suppose we have chosen labeled points so that only the nodes Valjean and Cosette are labeled but not Javert. Since the node Valjean has many more links than the node Cosette, the random walk starting from the node Valjean will less likely hit the node Woman 2 than the random walk starting from the node Cosette in some given time. Thus, the PageRank based method classifies the node Woman 2 into the class corresponding to Cosette. Since the node Woman 2 is just one link away from both Valjean and Cosette, the probability to hit these nodes before absorption is approximately equal. Thus, if we apply the Standard Laplacian method the classification will be determined by the expected number of returns to the labeled nodes before absorption. Since the labeled node Valjean lies in the larger and denser class, the Standard Laplacian method classifies the node Woman 2 into the class corresponding to Valjean. This example again confirms the fact that the Standard Laplacian method classifies border points into a large class and the PageRank based method on opposite classifies the border points into a small class.

Application to handwritten digits classification: We consider the classification of handwritten digits from '0' to '9' (USPS data set [49]). Class sizes are 1553, 1269, 929, 824,852, 716, 834, 792, 708, 821, respectively. We computed the weight matrix using Euclidean distance between the vectors of pixels and chose 5 k-nearest neighbours to make the similarity graph

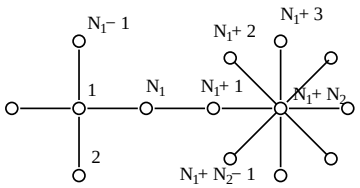
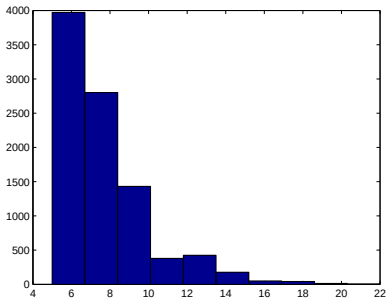
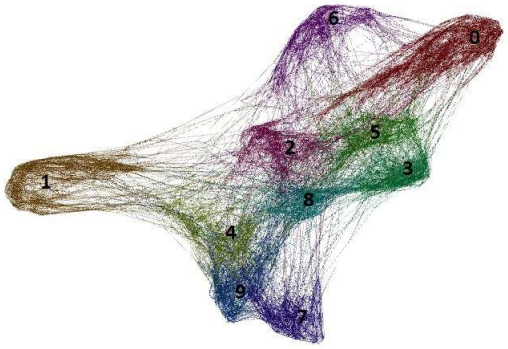


Figure 3.2: Characteristic network model.

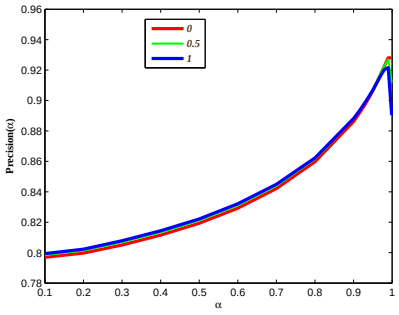
sparse and at the same time connected. Let us note that the similarity graph could be asymmetric after kNN, so we symmetrize it by making unweighted two-direction links. As a result we obtained a graph very close to regular (see the histogram in Figure 3.3(a)). We chose 5 labelled points for each class by max degree and randomly (averaged over 100 experiments). As one can see from the results, Figures 3.3(c)-(d), the curves are almost the same. This can be explained by our Corollary 3.1, as all data points and in particular labelled data points have nearly the same degree.



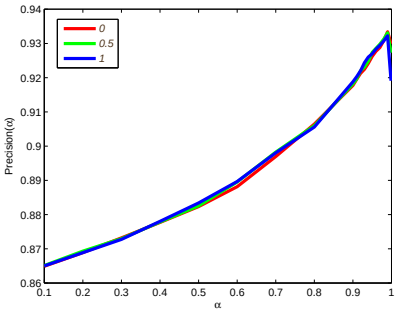
(a) Degree distribution.



(b) Similarity graph.



(c) Random Labelled Points.



(d) Max Degree Labelled Points.

Figure 3.3: Handwritten digits: Precision of the classification.

3.4 Conclusions and general recommendations

Using random walk theory, we provide insights about different graph-based semi-supervised learning methods. In particular, the Standard Laplacian method classifies border points into a large class and the PageRank based method on opposite classifies the border points into a small class. We also suggest the following general recommendations: If possible, choose labelled points with large degrees. Then, adopt the Standard Laplacian method with α in the upper-middle range of the interval $(0, 1)$. If finding large degree points is not feasible or recall is more important than precision for small classes, choose the PageRank based method.

4

SEMI-SUPERVISED METHODS FOR P2P CONTENT AND USER CLASSIFICATION

4.1 Introduction and summary of the results

P2P downloads still represent a large portion of today's Internet traffic. As of 2010, more than 100 million users operated BitTorrent and generated then more than 30% of the total Internet traffic [56]. According to the Wikipedia article about BitTorrent [2] (accessed in 2012), the traffic generated by BitTorrent was greater than the traffic generated by Netflix and Hulu combined. Of course, nowadays online video services like YouTube are more popular than P2P file download. We expect that the proposed semi-supervised learning methods if needed can also be adapted to classification of online video.

Recently, a significant research effort has been done to develop tools for automatic classification of Internet traffic by application [58, 57, 73]. The purpose of the present work is to provide a framework for subclassification of P2P traffic generated by the BitTorrent protocol. Unlike previous works [58, 57, 73], we cannot rely on packet level characteristics (packet size, packet interarrival time, etc). Instead we make use of the bipartite user-content graph. This is a graph formed by two sets of nodes: the set of users (peers) and the set of contents (downloaded files). From this basic bipartite graph we also construct the user graph, where two users are connected if they download the same content, and the content graph, where two files are connected if they are both downloaded by at least one user. Using methodology developed in [56] we were able to use the snapshots of P2P downloads from the whole Internet. Even a snapshot corresponding to half an hour duration represent a huge among of data. Without some filter-

ing technique, which will be explained later in the chapter we were even not able to operate with the user graph constructed from a single snapshot. The content graph is smaller and we were able to construct an aggregated content graph from several snapshots corresponding to the week-long observation.

The general intuition is that the users with similar interests download similar contents. This intuition can be rigorously formalized with the help of graph based semi-supervised learning approach. In particular, we have chosen to work with the three principal semi-supervised learning methods: the Standard Laplacian method, the Normalized Laplacian method and the PageRank based method. In the comprehensive comparative analysis [40] it has been shown that the Standard Laplacian method and the PageRank based methods are among the best performing semi-supervised learning methods. Thus, we concentrate in this chapter on these methods. It is also very important that these methods have implementations with quasi-linear time complexity and linear space complexity. This is a very important detail, as we deal in this chapter with large volumes of data. Some data sets used in the present chapter are several orders of magnitude larger than data sets typically used in the literature on graph based semi-supervised learning. One more goal of the present chapter is to validate the theoretical results from the previous chapter about the impact of the choice of the labelled nodes on classification result. In particular, we test the following three options for the choice of the labelled points: randomly chosen labelled points, labelled points with large degrees and labelled points with large PageRank value. We demonstrate that in the context of P2P classification the choice of labeled points with large degree or large PageRank values gives good results in the majority of classification tasks.

This chapter is mainly based on the articles:
K. Avrachenkov, P. Gonçalves, A. Legout and M. Sokol, “Classification of content and users in BitTorrent by Semi-supervised Learning methods”, In the IEEE IWCMC International Workshop on Traffic Analysis and Characterization, TRAC 2012
and
K. Avrachenkov, P. Gonçalves, A. Legout and M. Sokol, “Graph Based Classification of Content and Users in BitTorrent”, In the NIPS Big Learning Workshop, 2011.

4.2 Datasets and method implementation description

We have several snapshots of the Torrents collected from the whole Internet using methodology described in [56]. Each snapshot contains half an hour of P2P transfers. In total, we have about one week of observations. We have also an aggregate representing the transfers observed during the whole week. To test the effect of NATs, to save memory and to reduce information noise, the following filtering has been applied which we denote by $g(X, Y)$: we filter out all IP addresses with more than or equal to X ports (X = 0 means no filtering), and we filter out all contents with less than or equal to Y IP addresses seen downloading the content (Y = 0 means no filtering). Two users with the same IP addresses but with different ports could be the same user. So the filtering by ports helps us to reduce the influence of counting the same user as different ones. The second filter by IP address helps to remove unpopular contents which were downloaded less than or equals to Y times.

Table 4.1: The content graphs after preprocessing.

Graph	**# nodes**	**# edges**
g(2,10)	200 413	50 726 946
g(0,10)	200 487	174 086 752
g(2,0)	624 552	92 399 318

We use the whole aggregate to create the content graph. Some files are tagged with information about name, language, topic, login of the person who inserted these files. Those tags correspond to the classification made by popular torrent sites like ThePirateBay [56]. If two files are downloaded by the same user, we create an edge between these two files. The weight of the edge shows how many users downloaded these two files. We filter out all links with the weight equal to one to reduce the noise and memory usage. Without this filtering even the PageRank based method with quasi-linear complexity cannot be applied on a standard desktop computer.

We start with the smallest aggregated dataset $g(2, 10)$ which contain information with small noise. To evaluate the impact of the noise with respect to user identification we have also made experiments with datasets $g(0, 10)$ and $g(2, 0)$.

The graph for $g(2, 0)$ dataset after preprocessing contains three times more nodes and two times more edges than the dataset $g(2, 10)$. The graph for $g(0, 10)$ dataset after preprocessing contains two times more edges than the dataset $g(2, 10)$.

Let us now describe how we construct the user graph. The user graph is constructed with the help of HADOOP realization of MapReduce technology [1] from the basic user-content bipartite graph from a single half an hour snapshot. The aggregated user graph is too large to work with.

Table 4.2: The quantity of language base line expert classifications.

Language	#content	#user
English	36465	57632
Spanish	2481	2856
French	1824	2021
Italian	2450	3694
Japanese	720	416

Table 4.3: The quantity of topic base line expert classifications.

Topic	# content	# user
Audio Music	23639	13950
Video Movies	20686	43492
TV shows	12087	27260
Porn movies	8376	7082
App. Windows	4831	2874
Games PC	4527	8707
Books Ebooks	1185	281

The snapshot contains information on which content was downloaded by whom. In the user graph an edge with the weight M signifies that two users download M same files. The user graph has $3\,228\,410$ nodes and $3\,436\,442\,577$ edges. The number of edges with weight one is equal to $3\,309\,965\,972$. Also we have noticed that some users downloaded much more files than a normal user would do. One user who has downloaded $655\,727$ files for sure is a robot. Thus, we have decided remove all edges with weight one and the user-robot. The modified user graph has $1\,126\,670$ nodes and $124\,753\,790$ edges. This filtering significantly reduces required computing and memory resources. In fact, by doing this filtering we also remove some information noise. If two users download only one common item it could be by pure chance, if they both download more than two same files - it is more likely that they share same interests.

We classify contents and users by both language and topics. The considered languages and topics are given in Tables 4.2 and 4.3.

Our base line expert classification is based on P2P content tags if they are available. For instance, in the case of classification by language we consider that the content is in English if it has only tag “English”. And we consider a user to be an English language user, if he or she downloads only English language content.

We have implemented all the family of the proposed semi-supervised learning methods (methods corresponding to any value of parameter σ) in the WebGraph framework [26]. The

WebGraph framework has a very efficient graph compression technique which allows us to work with very large graphs.

4.3 Results of classification of content and users

Using PageRank based classification method, we have performed four classification experiments. We have used the aggregated graph of content $g(2,10)$ to classify the content into 5 classes according to the languages (see Table 4.2) and into 7 classes according to the content type (see Table 4.3). The classification of the aggregated content graph has taken approximately 15 minutes on a 64-bit computer with Intel-Core7i processor and 6GB RAM. The results of the classification evaluated in terms of accuracy are presented in Tables 4.4 and 4.5. Then, we have performed the classification of users also into 5 classes of the languages and into 7 classes of the content preferred by users (see Tables 4.6 and 4.7). It has taken about 20 minutes on the same computer. However, the preprocessing of a single snapshop of the user graph was much more demanding than the preprocessing of the aggregated content graph. Our main conclusion is that the PageRank based classification method scales remarkably well with large volumes of data. Then, our second important observation is that by using a very little amount of information, we are able to classify the content and users with high accuracy. For instance, in the dataset of $1\,126\,670$ users, using only 50 labelled nodes for each language, using the PageRank based method we are able to classify the users according to their preferred language with 88% accuracy.

In all four classification experiment, we have tried three different options for the choice of the labelled points. We have chosen the labelled points: (a) with largest standard PageRank values; (b) with largest degree; and (c) randomly. When evaluating the performance with the randomly chosen labelled points we have averaged the accuracy over 10 random samples (because of the size of the data, making more than 10 samples for each of many experimental setups was very time demanding) and we have also reported the worst (rand min column) and the best (rand max column) accuracy. With respect to the choice of the labelled points, our conclusion is that in the majority of cases the labelled points with large values of the standard PageRank are the best picks (see topPR columns). In the case of classification with the aggregated content graph, the labelled points with large degrees give results comparable with the results obtained with the labelled points chosen according to PageRank. However, it was interesting to observe that in the case of the classification of users, the classification based on the labelled points with large degrees does not perform well at all. Our explanation is that in that dataset the nodes with very large degrees are not representative. There is an independent confirmation of this idea given in [19].

We would like to note that there is not much difference if one considers weighted or unweighted graph for content classification. As one can see from Table 4.4, the accuracy of content classification by languages in the case of unweighted graph is 66.3% (choosing 50 labbelled points for each class according to top PageRank values). We have repeated the experiment with

the weighted graph and have obtain 68.9% accuracy. We explain the relatively small difference in accuracies by the fact that 88.3% of edges have weight 1 and then 7.1% of edges have weight 2. So the majority of edges have weight 1 and the other edges have also small weight.

Finally, we have observed that the classification using $g(2, 10)$ filtering is one or two percent better in terms of accuracy than the classification using $g(0, 10)$ filtering. Thus, by doing the filtering we not only reduce the amount of data required for processing, but also we reduce the information noise.

To understand better how the graph based semi-supervised learning works let us consider in the next two sections smaller subsets of content.

Table 4.4: Accuracy of the classifications for the $g(2, 10)$ dataset by languages.

# seeds	topPR	topDeg	rand (10Exp)	rand min	rand max
5	0.579	0.573	0.51	0.44	0.578
50	0.663	0.647	0.634	0.614	0.649
500	0.688	0.676	0.658	0.653	0.663

Table 4.5: Accuracy of the classifications for the $g(2, 10)$ dataset by topics.

# seeds	topPR	topDeg	rand(10Exp)	rand min	rand max
5	0.504	0.51	0.48	0.36	0.546
50	0.6344	0.6276	0.6278	0.604	0.645
500	0.7279	0.7182	0.6562	0.6525	0.6595

Table 4.6: Accuracy of the classifications for the user dataset by languages.

# seeds	topPR	topDeg	rand (10Exp)	rand min	rand max
5	0.788	0.765	0.732	0.613	0.817
50	0.88	0.78	0.834	0.82	0.85
500	0.853	0.535	0.901	0.896	0.907

Table 4.7: Accuracy of the classifications for the user dataset by topics.

# seeds	topPR	topDeg	rand(10Exp)	rand min	rand max
5	0.683	0.399	0.631	0.563	0.678
50	0.752	0.477	0.767	0.752	0.777
500	0.789	0.52	0.86	0.858	0.865

4.4 Classification of Video plus Music subgraph

We have constructed a subgraph which consists of all files which have in their tags “video”, “movie”, “audio” or “music”. In Table 4.8 we see the results of classification “music”+“audio” against “video”+“movie”. The results are quite good (accuracy 90% against accuracy 63.4% in the case of 50 labelled points chosen according to the top PageRank values, see Tables 4.5 and 4.8). The good classification is probably due to the fact that the dataset is smaller and the classes are balanced. In particular it is interesting to observe how the files tagged “Music Video Clips” are classified. 143 such files are classified into “music”, and 45 files are classified into “video”. This is quite in agreement with intuition that “music video clips” are better related to music than to video. On opposite only 20 “video movie clips” are classified as “music” and 125 “video movie clips” are classified as “video”. This also agrees with our intuition since most of “video movie clip” files are short extracts from movies.

# seeds	Accuracy	CV matrix		
50	0.90	music	30833	4337
topPR		video	4775	50862
500	0.938	music	32008	3162
topPR		video	2418	53219
1000	0.946	music	32705	2465
topPR		video	2474	53163
500m/1000v	0.942	music	32423	2747
topPR		video	2510	53127

Table 4.8: Accuracy and Cross-Validation (CV) matrix for music&audio vs video&movies classification, $\alpha = 0.5$.

4.5 Classification of untagged content

We have also created “other video” subgraph from the whole content graph. We have taken all nodes for which we have topic tags as “other video” and all edges induced by the supergraph. The subgraph contains 1189 nodes and 20702 edges. We made the expert evaluation manually by popular categories: “Sport Tutorials” [ST] (116), “Science Lectures” [SL] (127), “Japanese Cartoons” [JC] (93), “Porno” [P] (81), “Software Tutorials” [SFT] (113), “Movies” [M] (129).

The results of the semi-supervised classification are presented in Tables 4.9, 4.10, 4.11. In Table 4.9 we demonstrate the effect of the choice of the labelled points. As expected the more labelled points we take the better. In Table 4.12 we compare in detail the random choice of labelled points with the labelled points chosen according to their PageRank value. Specifically we average the results over 100 experiments with random labelled points. We can see that the precision corresponding to the labelled points chosen by PageRank is better than the average precision corresponding to the random choice of the labelled points. The coefficient of variation (CoV) for the random choice of the labelled points is significant (around 20%), which means that if we choose labelled point randomly the result of the classification is much less reliable than the result of the classification according to the labelled points with large PageRank values. It was surprising to observe that choosing labelled points with large degree does not help much. May be here we also face the phenomenon described in [19].

In Tables 4.10, 4.11 we present the Cross-Validation matrices for experiments with 10 and 15 labeled points chosen according to large PageRank values. In both tables we see strong diagonal domination. It is nice to observe that we have good classification despite the fact that nearly the half of the files do not belong to any of the mentioned above six classes. This can be interpreted as robustness of graph based semi-supervised learning approach with significant presence of noisy data.

Furthermore, it is interesting to observe that most of the “other video” files with the content as “Dance Tutorials” (21 from 27) are classified into “Sport Tutorials” [ST], which seems to be indeed related category. And all tutorials about gun shooting (13) are classified in “Sport Tutorials”, even though they have not initially been classified as “Sport Tutorials”. This automatic classification appears to be quite logical and suggests the possibility of application of graph based semi-supervised learning for refinement of P2P content categorization.

# seeds	TopPR	TopDeg	rand(100Exp)	rand min	rand max
1	0.56	0.519	0.45	0.21	0.64
5	0.66	0.53	0.62	0.53	0.7
10	0.70	0.66	0.685	0.623	0.73
15	0.731	0.68	0.72	0.66	0.75

Table 4.9: Accuracy for "Other Video" subgraph classification, $\alpha = 0.5$.

Classified as→	JC	M	P	SFT	SL	ST
JC	65	2	1	1	5	8
M	6	47	18	6	11	21
P	0	8	59	4	2	3
SFT	3	4	3	91	9	3
SL	5	5	3	10	85	19
ST	2	9	5	8	2	85

Table 4.10: Cross-Validation matrix for "Other Video" subgraph classification, TopPR 10 labeled points, $\alpha = 0.5$.

Classified as→	JC	M	P	SFT	SL	ST
JC	77	3	1	0	4	3
M	9	54	13	4	6	25
P	0	8	57	5	3	3
SFT	4	4	0	98	5	2
SL	5	7	2	9	92	12
ST	10	9	5	7	1	82

Table 4.11: Cross-Validation matrix for "Other Video" subgraph classification, TopPR 15 labelled points, $\alpha = 0.5$.

Seeds	Average	Variance	min	max	CoV
rand10	0.684	0.022	0.622	0.725	0.217
rand15	0.726	0.018	0.682	0.773	0.185

Table 4.12: Statistics for accuracy for "Other Video" subgraph classification, $\alpha = 0.5$, random labeled points, 100 experiments.

4.6 The effect of σ and α

In this section we invetigate the effect of σ and α on the quality of classification. We provide experiments for content and user graphs. For each class we have chosen 50 labelled points with the maximal degree and pagerank within the ground truth points. Since we do not have ground truth for all the points, it is assumed that choosing random points from the ground truth will not be representative (popular content is more likely to be tagged). The precision of classification for $\sigma = 0.0; 0.5; 1.0$ (Standard Laplacian, Normalized Laplacian and PageRank based methods) and various values of α is given in Figures 4.1 4.2.

The Figure 4.1 is consistent with Figure 3.1(b). We could classify content with precision more than 90% by languages and 70% by topics. In the case of language classification the Standard Laplacian method shows the best results and significantly overperforms the PageRank based method. But in case of topic classification PageRank method shows stable results and nearly as good as others methods. We could notice that the labelled points chosen according to PageRank values slightly increase precision in comparison with the case when the labelled points chosen according to maximal degree.

We could classify users (Figure 4.2) with precision more than 95% by languages and 80% by topics. Standard Laplacian method shows the best results and significantly outperforms the PageRank method by languages and by topics. The labelled points chosen according to PageRank values increase precision roughly by 5%. We obtained very interesting and rarely behaviour for PageRank based method. We noticed that at first PageRank attract more items from English language to others small classes. For Japanese and French languages it attracts much more items than for Spanish and Italian languages. But after some $\alpha > 0.95$ it unexpectedly starts to give back those items to English language class as shown in Tables 4.6.

We are glad to observe that the experiments of the current chapter confirm the theoretical results of the previous chapter. In particular, we maintain recommendation to choose the labelled points with large degree (or large values of PageRank) and use the Standard Laplacian method with the parameter α in the upper-middle range of the interval $(0, 1)$.

Class→	En	Fr	It	Jp	Sp
English	47201	2990	2461	4226	754
French	31	1965	9	15	1
Italian	29	2	3635	8	20
Japanese	48	21	5	342	0
Spanish	88	67	34	42	2625

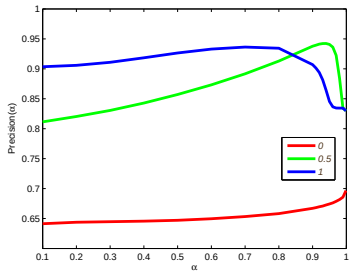
Table 4.13: Cross-Validation matrix for the user graph classification by languages, $\alpha = 0.1$, precision 83.71%.

Class→	En	Fr	It	Jp	Sp
English	37145	6263	2430	11049	745
French	4	2006	6	5	0
Italian	11	1	3658	8	16
Japanese	7	23	1	385	0
Spanish	27	39	27	45	2718

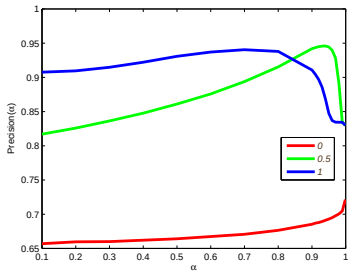
Table 4.14: Cross-Validation matrix for the user graph classification by languages, $\alpha = 0.95$, precision 68.92%.

Class→	En	Fr	It	Jp	Sp
English	48925	4585	1774	1209	1139
French	7	2006	5	2	1
Italian	19	1	3654	1	19
Japanese	18	27	5	365	1
Spanish	55	4	11	10	2776

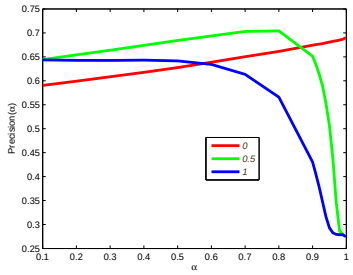
Table 4.15: Cross-Validation matrix for the user graph classification by languages, $\alpha = 0.999$, Precision 86.65%.



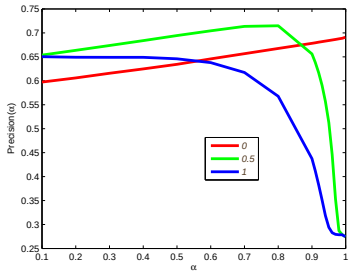
(a) Max Degree Labelled Points by Languages.



(b) Max PageRank Labelled Points by Languages.

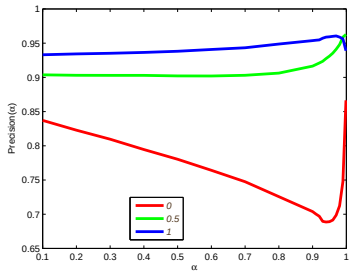


(c) Max Degree Labelled Points by Topics.

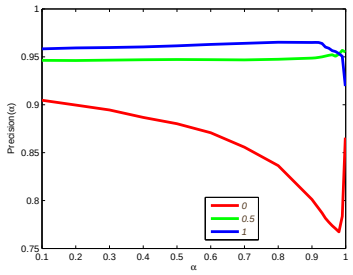


(d) Max PageRank Labelled Points by Topics.

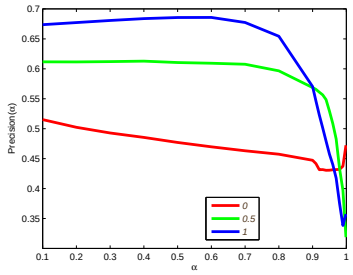
Figure 4.1: Content P2P Graph: Precision of the classification.



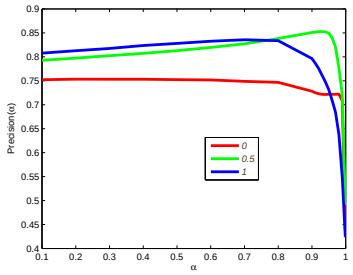
(a) Max Degree Labelled Points by Languages.



(b) Max PageRank Labelled Points by Languages.



(c) Max Degree Labelled Points by Topics.



(d) Max PageRank Labelled Points by Topics.

Figure 4.2: User P2P Graph: Precision of the classification.

4.7 Conclusions

We have proposed to apply the graph-based semi-supervised learning method to classify P2P content and users. The proposed methods have appeared to be highly scalable. We were able to deal with all world-wide torrents active in some point in time. With very few labelled points we have achieved very high precision. We have confirmed the theoretical conclusions of the previous chapter. One of our new recommendations is to choose labelled points with large values of PageRank. We have also demonstrated that the graph-based semi-supervised methods are very robust with respect to various types of noise in the data.

Part II

Quick Detection of Central Nodes in Complex Networks

5

QUICK DETECTION OF NODES WITH LARGE DEGREES

5.1 Introduction and summary of the results

Our goal in this chapter is to quickly find top k lists of nodes with the largest degrees in large complex networks. Firstly, node degree is one of centrality measures used for the analysis of complex networks. Secondly, large degree nodes can serve as proxies for central nodes corresponding to the other centrality measures as betweenness centrality or closeness centrality [79, 62]. Thirdly, as shown in Chapters 3-4, it is beneficial to select labelled data points with large degrees in semi-supervised learning methods.

In the present chapter we restrict ourselves to undirected networks or symmetrized versions of directed networks. In particular, this assumption is well justified in social networks. Typically, friendship or acquaintance is a symmetric relation. Also, in our applications for the semi-supervised learning methods we have considered undirected similarity graphs.

If the adjacency list of the network is known (not often the case in complex networks), a deterministic algorithm to find the top k list of nodes with the largest degrees requires an average complexity of $O(n)$, where n is the number of nodes in the network. For instance, if HeapSort is used to find the top k list of nodes with the largest degrees, the complexity estimation can be specified as $O(n + k\log(n))$. We assume that the degree is available when accessing a node (if this is not the case, the complexity should be counted in terms of links). However, even linear complexity can be very high for very large, possibly varying, complex networks. Furthermore, when crawling some online social networks like Facebook or Twitter, a

crawler is constrained by a certain limit on the speed of crawling. For example, Twitter has the limit of one access per minute for the rate of crawling for one standard account. Thus, to crawl the entire network with more than 500 million users we need more than 950 years. Certainly we would like to discover nodes with largest degrees well before the entire network is crawled.

In the present chapter we suggest using random walk based methods for detecting a small number of nodes with the largest degree. The main idea is that the random walk very quickly comes across large degree nodes. Thus, the analysis of our approach is equivalent to the analysis of hitting times of a random walk. In our numerical experiments random walks outperform the standard deterministic algorithms by orders of magnitude in terms of computational complexity. For instance, in our experiments with the web graph of the UK domain (about 18 500 000 nodes) the random walk method spends on average only about 5 400 steps to detect the largest degree node. Potential memory savings are also significant since the method does not require knowledge of the entire network. In many practical applications we do not need a complete ordering of the nodes and even, we can tolerate some errors in the top list of nodes. We observe that the random walk method obtains many nodes in the top list correctly and even those nodes that are erroneously placed in the top list have large degrees. Therefore, as typically happens in randomized algorithms [65, 68], we trade off exact results for very good approximate results or for exact results with high probability and gain significantly in computational efficiency.

The chapter is organized as follows: in the next section we introduce our basic random walk with uniform jumps and demonstrate that it is able to quickly find large degree nodes. Then, in Section 5.3 using configuration model we provide an estimate for the necessary number of steps for the random walk. In particular, we prove that our algorithm has sublinear complexity on the configuration network model. In Section 5.4 we propose stopping criteria that use very little information about the network. In Section 5.5 we show the benefits of allowing few erroneous elements in the top k list.

This chapter is mainly based on the articles:
K. Avrachenkov, N. Litvak, M. Sokol and D. Towsley, “Quick etection of nodes with large degrees”, In the 9th International Workshop on Algorithms and Models for the Web Graph, WAW 2012.
and
K. Avrachenkov, N. Litvak, M. Sokol and D. Towsley, “Quick detection of nodes with large degrees”, Accepted to *Internet Mathematics* journal. To appear in 2014.

5.2 Random walk with uniform jumps

Let us consider a random walk with uniform jumps which serves as a basic algorithm for quick detection of large degree nodes. The random walk with uniform jumps is described by the following transition probabilities [17]

$$p_{ij} = \begin{cases} \frac{\alpha/n+1}{d_i+\alpha}, & \text{if } i \text{ has a link to } j, \\ \frac{\alpha/n}{d_i+\alpha}, & \text{if } i \text{ does not have a link to } j, \end{cases} \tag{5.1}$$

where d_i is the degree of node i. The random walk with uniform jumps can be regarded as a random walk on a modified graph where all the nodes in the graph are connected by artificial edges with a weight α/n. The parameter α controls the rate of jumps. Introduction of jumps helps in a number of ways. As was shown in [17], it reduces the mixing time to stationarity. It also solves a problem encountered by a random walk on a graph consisting of two or more components, namely the inability to visit all nodes. The random walk with jumps also reduces the variance of the network function estimator [17]. This random walk resembles the PageRank random walk. However, unlike the PageRank random walk, the introduced random walk is time reversible. One important consequence of the reversibility of the random walk is that its stationary distribution is given by a simple formula

$$\pi_i(\alpha) = \frac{d_i + \alpha}{2|E| + n\alpha} \qquad \forall i \in V, \tag{5.2}$$

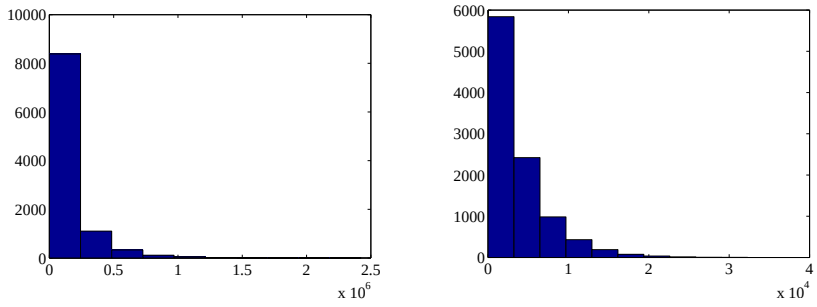
from which the stationary distribution of the unperturbed random walk can easily be retrieved. We observe that the modification preserves the monotonicity of the stationary distribution with respect to the node degree, which is particularly important for our application.

We illustrate on several network examples how the random walk helps us quickly detect large degree nodes. We consider as examples one synthetic network generated by the preferential attachment rule and two natural large networks. The Preferential Attachment (PA) network combines 100 000 nodes. It has been generated according to the generalized preferential attachment mechanism [37]. The average degree of the PA network is two and the power law exponent is 2.5. The first natural example is the symmetrized web graph of the whole UK domain crawled in 2002 [26]. The UK network has 18 520 486 nodes and its average degree is 28.6. The second natural example is the network of co-authorships of DBLP [23]. Each node represents an author and each link represents a co-authorship of at least one article. The DBLP network has 986 324 nodes and its average degree is 6.8.

We carry out the following experiment: we initialize the random walk (5.1) at a node chosen according to the uniform distribution and continue the random walk until we hit the largest degree node. The largest degrees for the PA, UK and DBLP networks are 138, 194 955, and 979, respectively. For the PA network we have made 10 000 experiments and for the UK

and DBLP networks we performed 1 000 experiments (these networks were too large to perform more experiments).

In Figue 5.1 we plot the histograms of hitting times for the PA network. The first remarkable observation is that when $\alpha = 0$ (no restart) the average hitting time, which is equal to 123 000 steps, is nearly three orders of magnitude larger than 3 720, the hitting time when $\alpha = 2$. The second remarkable observation is that 3 720 is of the same order of magnitude as the value $1/\pi_{max}(\alpha) = (2|E| + n\alpha)/(d_{max} + \alpha) = 2\,857$, which corresponds to the average return time to the largest degree node in the random walk with jumps.



(a) $\alpha = 0$ (b) $\alpha = 2$

Figure 5.1: Histograms of hitting times in the PA network.

We were not able to collect a representative number of experiments for the UK and DBLP networks when $\alpha = 0$. The reason for this is that the random walk gets stuck either in disconected or weakly connected components of the networks. For the UK network we were able to make 1 000 experiments with $\alpha = 0.001$ and obtain the average hitting time 30 750. Whereas if we take $\alpha = 28.6$ for the UK network, we obtain the average hitting time 5 800. Note that the expected return time to the largest degree node in the UK network is given by $1/\pi_{max}(\alpha) = (2|E| + n\alpha)/(d_{max} + \alpha) = 5\,432$. For the DBLP graph we conducted 1 000 experiments with $\alpha = 0.00001$ and obtained an average hitting time of 41 131. Whereas if we take $\alpha = 6.8$, we obtain an average hitting time of 14 200. The expected return time to the largest degree node in the DBLP network is given by $1/\pi_{max}(\alpha) = (2|E| + n\alpha)/(d_{max} + \alpha) = 13\,607$. The two natural network examples confirm our guess that the average hitting time for the largest degree node is fairly close to the average return time to the largest degree node, which is reciprocal to the value of the stationary distribution at the largest degree node. Next, using asymptotic analysis, we show that if α is sufficiently large, the principal term in the asymptotic expansion for the expected hitting time is close to the expected return time. Denote by H_j the hitting time to node j.

Theorem 5.1 *Without loss of generality, index the nodes such that node 1 is a node under consid-*

eration, $(1,i) \in E, i = 2,...,s, s = d_1 + 1$*, and let* ν *denote the initial distribution of the random walk with jumps. Then, for sufficiently large* α *and small* α/n*, the expected hitting time to node 1 starting from an arbitrary initial distribution* ν *is given by*

$$E_\nu[H_1] = \frac{\sum_{i=2}^{n} d_i + (n-1)\alpha}{d_1 + 2\alpha(1-1/n)} + O(1). \tag{5.3}$$

Proof: The expected hitting time from distribution ν to node 1 is given by the formula

$$E_\nu[H_1] = \nu[I - P_{-1}]^{-1}\underline{1}, \tag{5.4}$$

where P_{-1} is a taboo probability matrix (i.e., matrix P with the 1-st row and 1-st column removed). The matrix P_{-1} is substochastic but is very close to stochastic. Let us represent it as a stochastic matrix minus some perturbation term:

$$P_{-1} = \tilde{P} - \varepsilon Q = \tilde{P} - \begin{bmatrix} \frac{1+2\alpha/n}{d_2+\alpha} & 0 & & & & 0 \\ 0 & \ddots & & & & \\ & & \frac{1+2\alpha/n}{d_s+\alpha} & & & \\ & & & \frac{2\alpha/n}{d_{s+1}+\alpha} & & \\ & & & & \ddots & 0 \\ 0 & & & & 0 & \frac{2\alpha/n}{d_n+\alpha} \end{bmatrix}$$

We add missing probability mass to the diagonal of $\tilde{P}$, which corresponds to an increase in the weights for self-loops. The matrix $\tilde{P}$ represents a reversible Markov chain with the stationary distribution

$$\tilde{\pi}_j = \frac{d_j + \alpha}{\sum_{i=2}^{n} d_i + (n-1)\alpha}.$$

Now we can use the following result from the perturbation theory (see Lemma 1 in [9]):

$$[I - \tilde{P} + \varepsilon Q]^{-1} = \frac{\underline{1}\tilde{\pi}}{\tilde{\pi}(\varepsilon Q)\underline{1}} + X_0 + \varepsilon X_1 + \dots, \tag{5.5}$$

where $\tilde{\pi}$ is the stationary distribution of the stochastic matrix $\tilde{P}$. In our case, the quantity $\max_{i=2,...,s}\{1/(d_i+\alpha), 1/n\}$ will play the role of ε. We apply the series (5.5) to approximate the expected hitting time. Towards this goal, we calculate

$$\tilde{\pi}(\varepsilon Q)\underline{1} = \sum_{j=2}^{n} \tilde{\pi}_j \varepsilon q_{jj}$$

$$= \sum_{j=2}^{s} \frac{d_j + \alpha}{\sum_{i=2}^{n} d_i + (n-1)\alpha} \frac{1 + 2\alpha/n}{d_j + \alpha} + \sum_{j=s+1}^{n} \frac{d_j + \alpha}{\sum_{i=2}^{n} d_i + (n-1)\alpha} \frac{2\alpha/n}{d_j + \alpha}$$

$$= \frac{d_1(1 + 2\alpha/n) + (n - d_1 - 1)(2\alpha/n)}{\sum_{i=2}^{n} d_i + (n-1)\alpha} = \frac{d_1 + 2\alpha(1-1/n)}{\sum_{i=2}^{n} d_i + (n-1)\alpha}.$$

Observing that $\nu\underline{1}\tilde{\pi}\underline{1} = 1$, we obtain (5.3).

■

Indeed, the asymptotic expression (5.3) is very close to $(2|E| + n\alpha)/(d_1 + \alpha)$, which is the expected return time to node 1.

Based on the notion of the hitting time we propose an efficient method for quick detection of the top k list of largest degree nodes. The algorithm maintains a top k candidate list. Note that once one of the k nodes with the largest degrees appears in this candidate list, it remains there subsequently. Thus, we are interested in hitting events. We propose the following algorithm for detecting the top k list of largest degree nodes.

Algorithm 5.1 Random walk with jumps and candidate list

1. *Set* k, α *and* m.
2. *Execute a random walk step according to* (5.1). *If it is the first step, pick the initial node arbitrarily* (*in particular, the initial node can be chosen by the uniform distribution*).
3. *Check if the current node has a larger degree than one of the nodes in the current top* k *candidate list. If it is the case, insert the new node in the top-k candidate list and remove the worst node out of the list.*
4. *If the number of random walk steps is less than* m*, return to Step 2 of the algorithm. Stop, otherwise.*

The value of parameter α is not crucial. In our experiments, we have observed that as long as the value of α is neither too small nor too big, the algorithm performs well. According to our observations, a good option for the choice of α is a value around the average node degree. Let us explain this choice.

Consider a random walk $\{W_t\}_{t=0}^{\infty}$ with transition probabilities (5.1). We denote by $P_\nu(\cdot)$ the probability distribution of this Markov chain with initial distribution ν. Now assume that the Markov chain is in a stationary regime (the stationary regime is achieved quickly when the parameter α is not too small [17]). Then by the Bayes formula we derive two remarkable equations:

$$P_\pi[W_t = i|\text{jump}] = \frac{P_\pi[W_t = i, \text{jump}]}{P_\pi[\text{jump}]} = \frac{P_\pi[W_t = i]P_\pi[\text{jump}|W_t = i]}{\sum_{j=1}^{n} P_\pi[W_t = j]P_\pi[\text{jump}|W_t = j]}$$

$$= \frac{\frac{d_i+\alpha}{2|E|+n\alpha}\,\frac{\alpha}{d_i+\alpha}}{\sum_{j=1}^{n}\frac{d_j+\alpha}{2|E|+n\alpha}\,\frac{\alpha}{d_j+\alpha}} = \frac{1}{n}, \tag{5.6}$$

$$P_\pi[W_t = i|\text{no jump}] = \frac{P_\pi[W_t = i, \text{no jump}]}{P_\pi[\text{no jump}]}$$

$$= \frac{P_\pi[W_t = i]P_\pi[\text{no jump}|W_t = i]}{\sum_{j=1}^n P_\pi[W_t = j]P_\pi[\text{no jump}|W_t = j]} = \frac{\frac{d_i+\alpha}{2|E|+n\alpha}\,\frac{d_i}{d_i+\alpha}}{\sum_{j=1}^n \frac{d_j+\alpha}{2|E|+n\alpha}\,\frac{d_j}{d_j+\alpha}}$$

$$= \frac{d_i}{2|E|} = \pi_i(0), \quad i = 1, 2, \ldots, n. \tag{5.7}$$

Thus, in a stationary distribution, given that no jump occurred, the probability that $[W_t = i]$ is exactly $\pi_i(0)$!

Next observe that W_t is a regenerative process, where regeneration points are the jumps to the uniform distribution, and the regenerating cycles are independent. Concerning the choice of α, there is a clear trade-off: if α is too small, then regenerating cycles are long and a random walk can get entangled in some part of the network; but if α is too large, then the cycle will often consist only of one step corresponding to a jump. Thus, we would like to maximize the long-run fraction of independent observations from $\pi(0)$. To this end, we note that given m' cycles, the mean total number of steps is

$$m'E[\text{cycle length}] = m'(P_\pi[\text{jump}])^{-1}.$$

Out of the random walk run with m' cycles, m' independent observations from π are generated, from which on average $m'P_\pi[\text{jump}]$ observations coincide with a jump. As will be discussed in Section 4, we need to maximize the long-run fraction of independent observations, that are not a jump, in a sample compared to the number of steps of a random walk:

$$\frac{m' - m'P_\pi[\,\text{jump}]}{m'(P_\pi[\,\text{jump}])^{-1}} = P_\pi[\,\text{jump}](1 - P_\pi[\,\text{jump}]) \to \max.$$

Obviously, the maximum is achieved when

$$P_\pi[\text{jump}] = \frac{1}{2}.$$

It remains to rewrite $P_\pi[\text{jump}]$ in terms of the algorithm parameters:

$$P_\pi[\text{jump}] = \sum_{j=1}^n P_\pi[W_t = j]P_\pi[\text{jump}|W_t = j]$$

$$= \sum_{j=1}^n \frac{d_j + \alpha}{2|E| + n\alpha}\,\frac{\alpha}{d_j + \alpha} = \frac{n\alpha}{2|E| + n\alpha} = \frac{\alpha}{\bar{d} + \alpha}, \tag{5.8}$$

where $\bar{d} := 2|E|/n$ is the average degree. For the maximal efficiency, the last fraction above must be equal to $1/2$, which gives the optimal value for the parameter α

$$\alpha_* = \bar{d}.$$

With this choice of α, the random walk contains the maximal possible fraction of independent observations from the distribution $\pi_i(0)$.

The average degree is not necessarily known in advance. However, we may chose α based on our knowledge of samples of similar nature, and then estimate the average degree using (5.8) and the observed cycle length. Specifically, we can use the equation

$$E_u[T] = \frac{1}{P_\pi[\text{jump}]} = \frac{2|E|/n + \alpha}{\alpha}. \tag{5.9}$$

Then we can adjust α to its optimal value.

Theorem 5.1 demonstrates that the expected hitting time to a large degree node is approximately equal to the reciprocal of the stationary probability. Below we obtain an upper bound on the expected hitting time. Without loss of generality, let us consider node k from the top-k list ($d_1 \geq ... \geq d_k \geq d_{k+1} \geq ...$). Assume also that the initial node is chosen according to the uniform distribution. Let H_k be the hitting time to node k and let T be the time of the first jump (to the uniform distribution). Then, using Wald's identity, we can write

$$E_u[H_k] = E_u[\#\text{jumps on } [0, H_k]]E_u[\min\{T, H_k\}], \tag{5.10}$$

where $E_u[\cdot]$ is the expectation given the random walk starts from the uniform distribution. We note that

$$E_u[\min\{T, H_k\}] \leq E_u[T]. \tag{5.11}$$

Next, we also note that

$$E_u[\#\text{jumps on } [0, H_k]] = \frac{1}{P_u[H_k \leq T]}. \tag{5.12}$$

Next, we provide a lower bound for the probability $P_u[H_k \leq T]$. This lower bound give a good approximation, if we assume that node k is usually found within the first two steps of a cycle. This is a natural assumption, if α is not too small, and consequently, the cycles are not too large. In particular, this is the case if we choose the value of α as the average degree. Then, we have

$$\begin{aligned} P_u[H_k \leq T] \geq P_u[H_k \leq \min\{T, 2\}] &= \frac{1}{n} + \frac{1}{n} \sum_{i:(i,k)\in E} \frac{\alpha/n + 1}{d_i + \alpha} \\ &> \frac{d_k}{n} \cdot \frac{1}{d_k} \sum_{i:(i,k)\in E} \frac{\alpha/n + 1}{d_i + \alpha} \geq \frac{d_k}{n} \cdot \frac{\alpha/n + 1}{d_k^{-1} \sum_{i:(i,k)\in E} d_i + \alpha}. \end{aligned} \tag{5.13}$$

Combining the above equation with (5.9)–(5.12), we obtain

$$E_u[H_k] \leq \frac{n}{d_k} \cdot \frac{\bar{d} + \alpha}{\alpha} \cdot \frac{d_k^{-1} \sum_{i:(i,k)\in E} d_i + \alpha}{\alpha/n + 1}. \tag{5.14}$$

In particular, choosing $\alpha = \bar{d}$ in (5.14) yields

$$E_u[H_k] \leq \frac{2n}{d_k} \cdot \frac{d_k^{-1} \sum_{i:(i,k)\in E} d_i + \bar{d}}{\bar{d}/n + 1}. \tag{5.15}$$

The number of random walk steps, m, is a crucial parameter. Our experiments indicate that we obtain a top k list with many correct elements with high probability if we take the number of random walk steps to be twice or thrice as large as the expected hitting time of the nodes in the top k list. This observation can be made rigorous thanks to the result from [27, Ch.9,p.333] that we can adapt for our situation as follows.

Proposition 5.1 *Let* $H_1, ..., H_k$ *denote the hitting times to the top-*k *nodes with the largest degrees* $(d_1 \geq ... \geq d_k \geq d_{k+1} \geq ...)$. *Then, the expected time,* $E_u[\tilde{H}]$*, for the random walk with transition probabilities* (5.1) *and starting from the uniform distribution to detect a fraction* β *of top-*k *nodes is bounded by*

$$E_u[\tilde{H}] \leq \frac{1}{1-\beta} E_u[H_k]. \tag{5.16}$$

From Theorem 1 or bound (5.15), we know that the expected hitting time of a large degree node is related to the value of the node's degree. Thus, the problem of choosing m reduces to the problem of estimating the values of the largest degrees. We address this problem in the following section.

5.3 Estimating the largest degrees in the configuration network model

The estimations for the values of the largest degrees can be derived in the configuration network model [46] with a power law degree distribution. In some applications the knowledge of the power law parameters might be available to us. For instance, it is known that web graphs have power law degree distribution and we know typical ranges for the power law parameters (see e.g., [20]).

We assume that the node degrees $D_1, \ldots, D_n$ are i.i.d. random variables with a power law distribution F and finite expectation $E[D]$. Let us determine the number of links contained in the top k nodes. Denote

$$F(x) = P[D \leq x], \quad \bar{F}(x) = 1 - F(x), \quad x \geq 0.$$

Further let $D_{(1)} \geq \ldots \geq D_{(n)}$ be the order statistics of $D_1, \ldots, D_n$. Under the assumption that D_j's obey a power law, we use the results from the extreme value theory as presented in [64], to state that there exist sequences of constants (a_n) and (b_n) and a constant δ such that

$$\lim_{n\to\infty} n\bar{F}(a_n x + b_n) = (1 + \delta x)^{-1/\delta}. \tag{5.17}$$

This implies the following approximation for high quantiles of F, with exceedance probability close to zero [64]:

$$x_p \approx a_n \frac{(pn)^{-\delta} - 1}{\delta} + b_n.$$

For the jth largest degree, where $j = 2, \ldots, k$, the estimated exceedance probability equals $(j-1)/n$, and thus we can use the quantile $x_{(j-1)/n}$ to approximate the degree $D_{(j)}$ of this node:

$$D_{(j)} \approx a_n \frac{(j-1)^{-\delta} - 1}{\delta} + b_n. \tag{5.18}$$

The sequences (a_n) and (b_n) are easy to find for a given shape of the tail of F. Below we derive the corresponding results for the commonly accepted Pareto tail distribution of D, that is,

$$\bar{F}(x) = Cx^{-\gamma} \quad \text{for } x > x', \tag{5.19}$$

where $\gamma > 1$ and x' is a fixed sufficiently large number so that the power law degree distribution is observed for nodes with degree larger than x'. In that case we have

$$\lim_{n\to\infty} n\bar{F}(a_n x + b_n) = \lim_{n\to\infty} nC(a_n x + b_n)^{-\gamma}$$

$$= \lim_{n\to\infty} (C^{-1/\gamma} n^{-1/\gamma} a_n x + C^{-1/\gamma} n^{-1/\gamma} b_n)^{-\gamma},$$

which directly gives (5.17) with

$$\delta = 1/\gamma, \quad a_n = \delta C^\delta n^\delta, \quad b_n = C^\delta n^\delta. \tag{5.20}$$

Substituting (5.20) into (5.18) we obtain the following prediction for $D_{(j)}$, $j = 2, \ldots, k$, in the case of the Pareto tail of the degree distribution:

$$D_{(j)} \approx C^{1/\gamma}(j-1)^{-1/\gamma} n^{1/\gamma}. \tag{5.21}$$

It remains to find an approximation for $D_{(1)}$, the maximal degree in the graph. From the extreme value theory it is well known that if $D_1, \ldots, D_n$ obey a power law then

$$\lim_{n\to\infty} P\left(\frac{D_{(1)} - b_n}{a_n} \le x\right) = H_\delta(x) = \exp(-(1+\delta x)^{-1/\delta}),$$

where, for Pareto tail, a_n, b_n and δ are defined in (5.20). Thus, as an approximation for the maximal node degree we can choose $a_n x + b_n$ where x can be chosen as either a mean, a median or a mode of $H_\delta(x)$. If we choose the mode, $((1+\delta)^{-\delta} - 1)/\delta$, then we obtain an approximation, which is smaller than the one for the 2nd largest degree. Further, the mean $(\Gamma(1-\delta) - 1)/\delta$ is very sensitive to the value of $\delta = 1/\gamma$, especially when γ is close to one, which is often the case in complex networks. Besides, the parameter γ is hard to estimate with high precision. Thus, we suggest to choose the median $(\log(2))^{-\delta} - 1)/\delta$, which is less sensitive to the value of δ. This yields

$$D_{(1)} \approx a_n \frac{(\log(2))^{-\delta} - 1}{\delta} + b_n = C^{1/\gamma}(\log(2))^{-1/\gamma} n^{1/\gamma}. \tag{5.22}$$

For instance, in the PA network $\gamma = 2.5$ and $C = 3.7$, which gives according to (5.22) $D_{(1)} \approx 195$. (This is a reasonably good prediction even though the PA network is not generated according to the configuration model. We also note that even though the extremum distribution in the preferential attachment model is different from that of the configuration model their ranges seem to be quite close [67].) This in turn suggests that for the PA network m should be chosen in the range 6 000-18 000 if $\alpha = 2$. As we can see from Figure 5.2 this is indeed a good range for the number of random walk steps. In the UK network $\gamma = 1.7$ and $C = 90$, which gives $D_{(1)} \approx 329\,820$ and suggests a range of 20 000-30 000 for m if $\alpha = 28.6$. Figure 5.3 confirms that this is a good choice. The degree distribution of the DBLP network does not follow a power law so we cannot apply the above reasoning to it.

We conclude this section with a remark that from equation (5.21), bound (5.15) and Proposition 5.1, it follows that we can find a β fraction of top-k largest degree nodes in sublinear expected time in the configuration model. That is, we have

$$E_u[\tilde{H}] \le \frac{2}{1-\beta} \cdot \frac{d_k^{-1} \sum_{i:(i,k)\in E} d_i + \bar{d}}{\bar{d}/n + 1} \cdot \frac{n}{C^{1/\gamma}(k-1)^{-1/\gamma} n^{1/\gamma}}.$$

In particular, the last fraction above is of the order $\tilde{C}n^{\frac{\gamma-1}{\gamma}}$. If γ is close to one (which is often the case in complex networks), the computational savings compared to the deterministic approach can be very significant. For instance, for the UK network with $k = 10$ and $\beta = 0.8$, the bound (5.16) gives

$$E_u[\tilde{H}] \leq 214150,$$

which means at least 86-fold computational savings.

5.4 Stopping criteria

Suppose now that we do not have any information about the range for the largest k degrees. In this section we design stopping criteria that do not require knowledge about the structure of the network. As we shall see, knowledge of the order of magnitude of the average degree might help, but this knowledge is not imperative for a practical implementation of the algorithm.

Let us now assume that node j can be sampled independently with probability $\pi_j(\alpha)$ as in (5.2). There are at least two ways to achieve this practically. The first approach is to run the random walk for a significant number of steps until it reaches the stationary distribution. If one chooses α reasonably large, say the same order of magnitude as the average degree, then the mixing time becomes quite small [17] and we can be sure to reach the stationary distribution in a small number of steps. Then, the last step of a run of the random walk will produce an i.i.d. sample from a distribution very close to (5.2). The second approach is to run the random walk uninterruptedly, also with a significant value of α, and then perform Bernoulli sampling with probability q after a small initial transient phase. If q is not too large, we shall have nearly independent samples following the stationary distribution (5.2). In our experiment, $q \in [0.2, 0.5]$ gives good results when α has the same order of magnitude as the average degree.

We now estimate the probability of detecting correctly the top k list of nodes after m i.i.d. samples from (5.2). Denote by X_i the number of hits at node i after m i.i.d. samples. We note that if we use the second approach to generate i.i.d. samples, we spend approximately m/q steps of the random walk. We correctly detect the top k list with the probability given by the multinomial distribution

$$P[X_1 \geq 1, ..., X_k \geq 1] =$$

$$\sum_{i_1 \geq 1, ..., i_k \geq 1} \frac{m!}{i_1! \cdots i_k! (m - i_1 - ... - i_k)!} \pi_1^{i_1} \cdots \pi_k^{i_k} (1 - \sum_{i=1}^{k} \pi_i)^{m - i_1 - ... - i_k}$$

but it is not feasible for any realistic computations. Therefore, we propose to use the Poisson approximation. Let Y_j, $j = 1, ..., n$ be independent Poisson random variables with means $\pi_j m$. That is, the random variable Y_j has the following probability mass function $P[Y_j = r] = e^{-m\pi_j}(m\pi_j)^r / r!$. It is convenient to work with the complementary event of not detecting correctly the top k list. Then, we have

$$P[\{X_1 = 0\} \cup ... \cup \{X_k = 0\}] \leq 2P[\{Y_1 = 0\} \cup ... \cup \{Y_k = 0\}]$$

$$= 2(1 - P[\{Y_1 \geq 1\} \cap ... \cap \{Y_k \geq 1\}]) = 2(1 - \prod_{j=1}^{k} P[\{Y_j \geq 1\}])$$

$$= 2(1 - \prod_{j=1}^{k} (1 - P[\{Y_j = 0\}])) = 2(1 - \prod_{j=1}^{k} (1 - e^{-m\pi_j})) =: a, \tag{5.23}$$

where the first inequality follows from [65, Thm 5.10]. In fact, in our numerical experiments we observed that the factor 2 in the first inequality is very conservative. For large values of m, the Poisson bound without 2 works very well as proper approximation.

For example, if we would like to obtain the top 10 list with at most 10% probability of error, we need to have on average 4.5 hits per each top element. This can be used to design the stopping criteria for our random walk algorithm. Let $\bar{a} \in (0,1)$ be the admissible probability of an error in the top k list. Now the idea is to stop the algorithm after m steps when the estimated value of a for the first time is lower than the critical number $\bar{a}$. Clearly,

$$\hat{a}_m = 2(1 - \prod_{j=1}^{k}(1 - e^{-X_j}))$$

is the maximum likelihood estimator for a, so we would like to choose m such that $\hat{a}_m \leq \bar{a}$. The problem, however, is that we do not know which X_j's are the realisations of the number of visits to the top k nodes. Then let $X_{j_1}, ..., X_{j_k}$ be the number of hits to the current elements in the top k candidate list and consider the estimator

$$\hat{a}_{m,0} = 2(1 - \prod_{i=1}^{k}(1 - e^{-X_{j_i}})),$$

which is the maximum likelihood estimator of the quantity

$$2(1 - \prod_{i=1}^{k}(1 - e^{-m\pi_{j_i}})) \geq a.$$

(Here π_{j_i} is a stationary probability of the node with the score X_{j_i}, $i = 1, \ldots, k$). The estimator $\hat{a}_{m,0}$ is computed without knowledge of the top k nodes or their degrees, and it is an estimator of an upper bound of the estimated probability that there are errors in the top k list. This leads to the following stopping rule.

Stopping rule 0. *Stop at* $m = m_0$*, where*

$$m_0 = \arg\min\{m : \hat{a}_{m,0} \leq \bar{a}\}.$$

The above stopping criterion can be simplified even further to avoid computation of $\hat{a}_{m,0}$. Since

$$\hat{a}_{m,1} := 2(1 - (1 - e^{-X_{j_k}})^k) \geq \hat{a}_{m,0} \geq \hat{a},$$

where X_{j_k} is the number of hits of the worst element in the candidate list. The inequality $\hat{a}_m \leq \bar{a}$ is guaranteed if $\hat{a}_{m,1} \leq \bar{a}$. This leads to the following stopping rule for the random walk algorithm.

Stopping rule 1. *Compute* $x_0 = \arg\min\{x \in \mathbb{N} : (1 - e^{-x})^k \geq 1 - \bar{a}/2.\}$ *Stop at*

$$m_1 = \arg\min\{m : X_{j_k} = x_0\}.$$

We have observed in our numerical experiments that we obtain the best trade off between the number of steps of the random walk and the accuracy if we take α around the average degree and the sampling probability q around 0.5. Specifically, if we take $\bar{a}/2 = 0.15$ ($x_0 = 4$) in Stopping rule 1 for top 10 list, we obtain 87% accuracy for an average of 47 000 random walk steps for the PA network; 92% accuracy for an average of 174 468 random walk steps for the DBLP network; and 94% accuracy for an average of 247 166 random walk steps for the UK network. We have averaged over 1000 experiments to obtain tight confidence intervals.

5.5 Relaxation of top k lists

In the stopping criteria of the previous section we have strived to detect all nodes in the top k list. This costs us a lot of steps of the random walk. We can significantly gain in performance by relaxing this strict requirement. For instance, we could just ask for list of k nodes that contains 80% of top k nodes [16]. This way we can take an advantage of a generic 80/20 rule that 80% of result can be achieved with 20% of effort.

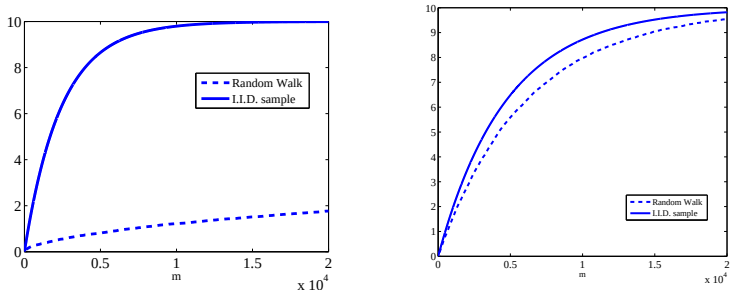
Let us calculate the expected number of top k elements observed in the candidate list up to trial m. Define by X_j the number of times we have observed node j after m trials and

$$H_j = \begin{cases} 1, & \text{node } j \text{ has been observed at least once,} \\ 0, & \text{node } j \text{ has not been observed.} \end{cases}$$

Assuming we sample in i.i.d. fashion from the distribution (5.2), we can write

$$\begin{aligned} E[\sum_{j=1}^{k} H_j] &= \sum_{j=1}^{k} E[H_j] = \sum_{j=1}^{k} P[X_j \geq 1] \\ &= \sum_{j=1}^{k} (1 - P[X_j = 0]) = \sum_{j=1}^{k} (1 - (1 - \pi_j)^m). \end{aligned} \tag{5.24}$$

In Figure 5.2 we plot $E[\sum_{j=1}^{k} H_j]$ (the curve "I.I.D. sample") as a function of m for $k = 10$ for the PA network with $\alpha = 0$ and $\alpha = 2$. In Figure 5.3 we plot $E[\sum_{j=1}^{k} H_j]$ as a function of m for $k = 10$ for the UK network with $\alpha = 0.001$ and $\alpha = 28.6$. The results for the UK and DBLP networks are similar in spirit.

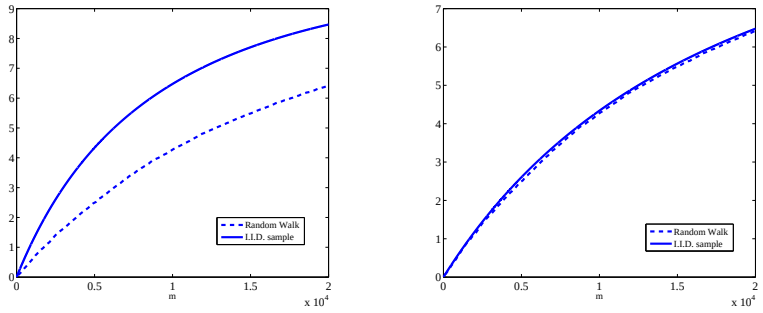


(a) $\alpha = 0$ (b) $\alpha = 2$

Figure 5.2: Average number of correctly detected elements in top-10 for PA.

Here again we can use the Poisson approximation

$$E[\sum_{j=1}^{k} H_j] \approx \sum_{j=1}^{k} (1 - e^{-m\pi_j}).$$



(a) $\alpha = 0.001$ (b) $\alpha = 28.6$

Figure 5.3: Average number of correctly detected elements in top-10 for UK.

In fact, the Poisson approximation is so good that if we plot it on Figures 5.2 and 5.3, it nearly covers exactly the curves labeled "I.I.D. sample", which correspond to the exact formula (5.24). Similarly to the previous section, we can propose stopping criteria based on the Poisson approximation. Denote

$$b_m = \sum_{i=1}^{k} (1 - e^{-X_{j_i}}).$$

Stopping rule 2. *Stop at* $m = m_2$*, where*

$$m_2 = \arg\min\{m : b_m \geq \bar{b}\}.$$

Now if we take $\bar{b} = 7$ in Stopping rule 2 for top-10 list, we obtain on average 8.89 correct elements for an average of 16 725 random walk steps for the PA network; we obtain on average 9.28 correct elements for an average of 66 860 random walk steps for the DBLP network; and we obtain on average 9.22 correct elements for an average of 65 802 random walk steps for the UK network. (We have averaged over 1000 experiments for each network.) This makes for the UK network the gain of more than two orders of magnitude in computational complexity with respect to the deterministic algorithm.

5.6 Conclusions

We have proposed the random walk method with the candidate list for quick detection of largest degree nodes and analyzed the complexity of the method by means of random walk hitting times. Using Poisson approximation, we have supplied stopping criteria which do not require knowledge of the graph structure. In the case of large networks, our algorithm finds top-k list of largest degree nodes with few mistakes with the running time orders of magnitude faster than the deterministic algorithms. In fact, for the configuration network model we have proved that our algorithm has sublinear time complexity.

6

QUICK DETECTION OF NODES WITH LARGE VALUES OF PAGERANK

6.1 Introduction

Personalized PageRank (PPR) or Topic-Sensitive PageRank [45] is a generalization of PageRank [72], and is a stationary distribution of a random walk on an entity graph, with random restart from a given personalization distribution. Originally designed for personalization of the Web search results [45], PPR found a large number of network applications, e.g., in finding related entities [32], graph clustering and finding local cuts [8, 10], link predictions in social networks [59] and protein-protein interaction networks [77]. The application of PPR to the person name disambiguation problem led to the first official place in the WePS 2010 challenge [75]. In most of applications, e.g., in name disambiguation, one is mainly interested in detecting top-k elements with the largest PPR and not in determining the exact values of PPR. This work on detecting top-k elements is driven by the following two key observations:

Observation 1: Often it is extremely important to detect fast the top-k elements with the largest PPR, while the exact order in the top-k list as well as the exact values of the PPR are by far not so important. Application examples are given in the above mentioned references.

Observation 2: We may apply a relaxation that allows a small number of elements to be placed erroneously in the top-k list. If the PPR values of these elements are of a similar order of magnitude as in the top-k list, then such relaxation does not affect applications, but it enables

us to take advantage of the generic "80/20 rule": 80% of the result is achieved with 20% of efforts.

We claim that the Monte Carlo approach naturally takes into account the two key observations. In [30] this approach was proposed for the computation of the standard PageRank. The estimation of the convergence rate in [30] was very pessimistic. The implementation of the Monte Carlo approach was improved in [39] and also applied there to PPR. Both [30] and [39] only use end points of the random walks to compute the PageRank values. Moreover, [39] requires extensive precomputation efforts and is very demanding in storage resource. In [18] the authors have further improved the realization of the Monte Carlo method [39]. In [14] it is shown that Monte Carlo estimation for large PageRank values requires about the same number of operations as one iteration of the power iteration method. In this chapter we show that the Monte Carlo algorithms require an incomparably smaller number of operations when our goal is to detect a top-k list with k not large. In our test on the Wikipedia entity graph with about 2 million nodes typically few thousands of operations are enough to detect the top-10 list with just two or three erroneous elements. Hence, we obtain a relaxation of the top-10 list with just about 1-5% of operations required by one power iteration. Experimental results on the Web graph appear to be even more striking. In the present work we provide theoretical justifications for such remarkable efficiency. We would like to emphasize that the Monte Carlo approach allows easy online and parallel implementation and does not require the knowledge of the complete graph.

This chapter is mainly based on the article:
K. Avrachenkov, N. Litvak, D. Nemirovsky, E. Smirnova and M. Sokol, "Quick detection for top-k Personalized PageRank lists", In the 8th International Workshop on Algorithms and Models for the Web Graph, WAW 2011.

6.2 Monte Carlo methods

Given a directed or undirected graph connecting some entities, the Personalized PageRank $\mathrm{ppr}(s, c)$ with a seed node s and a damping parameter α is defined as a solution of the following equations

$$\mathrm{ppr}(s, \alpha) = \alpha \mathrm{ppr}(s, \alpha) P + (1 - \alpha) \mathbf{1}_s^T, \qquad \sum_{j=1}^{n} \mathrm{ppr}_j(s, \alpha) = 1,$$

where $\mathbf{1}_s^T$ is a row unit vector with one in the sth entry and all the other elements equal to zero, P is the transition matrix associated with the entity graph and n is the number of entities. Equivalently, PPR can be given by [55]

$$\mathrm{ppr}(s, \alpha) = (1 - \alpha) \mathbf{1}_s^T [I - \alpha P]^{-1}. \tag{6.1}$$

When the values of s and α are clear from the context we shall simply write ppr.

We note that PPR is often defined with a general distribution v in place of $\mathbf{1}_s^T$. However, typically v has a small support. Then, due to linearity, the problem of PPR with distribution v reduces to computing PPR with distribution $\mathbf{1}_s^T$ [50].

In this work we consider two Monte Carlo algorithms. The first algorithm is inspired by the following observation. Consider a random walk $\{X_t\}_{t \geq 0}$ that starts from node s, i.e, $X_0 = s$. Let at each step the random walk terminate with probability $1 - \alpha$ and make a transition according to the matrix P with probability α. Then, the end-points of such a random walk has the distribution $\mathrm{ppr}(s, \alpha)$.

Algorithm 6.1 (MC End Point) *Simulate m runs of the random walk $\{X_t\}_{t \geq 0}$ initiated at node s. Evaluate ppr_j as a fraction of m random walks which end at node $j \in 1, \ldots, n$.*

Next, we exploit the fact that the element (s, j) of the matrix $[I - \alpha P]^{-1}$ equals to the expected number of visits to node j by the random walk initiated at state s with the run time geometrically distributed with parameter c [14]. Thus, the formula (6.1) suggests the following estimator for the PPR

$$\hat{\mathrm{ppr}}_j(s, \alpha) = (1 - \alpha) \frac{1}{m} \sum_{r=1}^{m} N_j(s, r), \tag{6.2}$$

where $N_j(s, r)$ is the number of visits to state j during the run r of the random walk initiated at node s and m is the number of runs. This leads to our second Monte Carlo algorithm.

Algorithm 6.2 (MC Complete Path) *Simulate m runs of the random walk $\{X_t\}_{t \geq 0}$ initiated at node s. Evaluate ppr_j as the total number of visits to node j multiplied by $(1 - \alpha)/m$.*

As outputs of the proposed algorithms we would like to obtain with high probability either a *top-k list* of nodes or a *top-k basket* of nodes.

Definition 6.1 *The* top-k list *of nodes is a list of* k *nodes with largest PPR values arranged in a descending order of their PPR values.*

Definition 6.2 *The* top-k basket *of nodes is a set of* k *nodes with largest PPR values with no ordering required.*

It turns out that it is beneficial to relax our goal and to obtain a top-k basket with a small number of erroneous elements.

Definition 6.3 *We call* relaxation-l *top-*k *basket a realization when we allow at most* l *erroneous elements from top-*k *basket.*

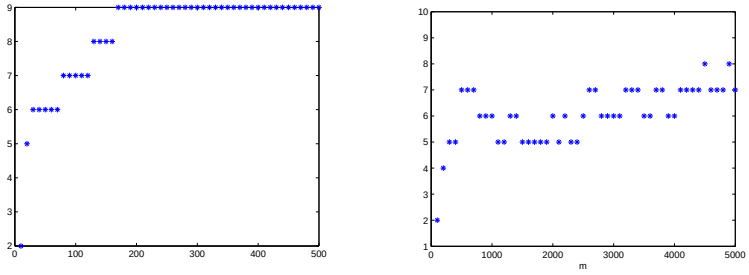
In the present work we aim to estimate the numbers of random walk runs m sufficient for obtaining top-k list or top-k basket or relaxation-l top-k basket with high probability. In particular, we demonstrate that ranking converges considerably faster than the values of PPR and that a relaxation-l with quite small l helps significantly.

Throughout the chapter we illustrate the theoretical analysis with the help of experiments on two large graphs: the Wikipedia entity graph and the Web graph. There is a number of reasons why we have chosen the Wikipedia entity graph. Firstly, all elements of PPR can be computed with high precision for the Wikipedia entity graph with the help of BVGraph/WebGraph framework [26]. Secondly, the Wikipedia graph has already been used in several applications related to finding top-k semantically related entities. Thirdly, since the Wikipedia entity graph has a very small average distance [86], it represents a very challenging test for the Monte Carlo methods. In just 3-4 steps the random walk can be very far from the starting node. Since the Monte Carlo approach does not require the knowledge of a complete graph, we can apply our algorithms to the actual Web graph. However, computing the exact values for the Pesonalized PageRank of web pages is infeasible in our experiments. We can only obtain correct top-k lists by Monte Carlo methods with very high probability as in [14, 39] using an ample number of crawls.

Illustrating example with Wikipedia: Following our recent work [75] we illustrate PPR by application to the person name disambiguation problem. One of the most common English names is Jackson. We have selected three Jacksons who have entries in Wikipedia: Jim Jackson (ice hockey), Jim Jackson (sportscaster) and Michael Jackson. Two Jacksons have even a common given name and both worked in ice hockey, one as an ice hockey player and another as an ice hockey sportscaster. In [15] we provide the exact lists of top-10 Wikipedia articles arranged according to PPR vectors. We observe that an exact top-10 list identifies quite well its seed node. Next, we run the Monte Carlo End Point method starting from each seed node. Notice that to obtain a relaxed top-10 list with two or three erroneous elements we need different number of

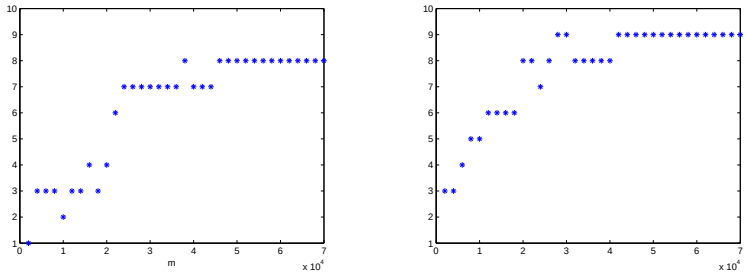
runs for different seed nodes (50000 runs for `Michael Jackson` vs. 500 runs for `Jim Jackson (ice hockey)`). Intuitively, the more immediate neighbours a node has, the larger number of Monte Carlo steps is required. Indeed, if a seed node has many immediate neighbours then the Monte Carlo method easily drifts away. In Figures 6.1-6.2.(a) we present examples of typical runs of the Monte Carlo End Point method for the three different seed nodes. An example of the Monte Carlo Complete Path method for the seed node `Michael Jackson` is given in Figure 6.2.(b). As expected, it outperforms the Monte Carlo End Point method. In the following sections we shall quantify all the above qualitative observations.

Illustrating example with the Web: We have also tested our two Monte Carlo methods on the Web. To see the difference in comparison with a "smaller" Wikipedia graph we have chosen the official Web page of Michael Jackson `http://www.michaeljackson.com` and the Web page of the hockey player statistics Jim Jackson hosted at `http://www.hockeydb.com`. In Figures 6.3.(a) and 6.3.(b) we present examples of typical runs of the Monte Carlo Complete Path and End Point methods for, respectively, the Michael Jackson Web page and the Jim Jackson Web page as a seed node. We have performed enough steps (6×10^5) to make sure that the top-k lists of nodes are stabilized so that we could say with very high certainty that we know the correct top-k lists. We observe that in comparison to the Wikipedia graph we need longer runs. However, the amount of computational saving is still very impressive. Indeed, according to even modest estimates, the size of the Web is more than 10^{10} pages. However, to get a good top-k list for the Michael Jackson page we need about 10^5 steps with MC Complete Path. Thus, we are using only 10^{-5} fraction of computational resources which are needed for just one power iteration!



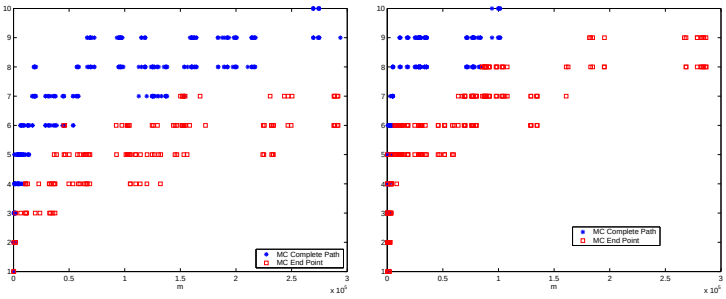
(a) Seed node `Jim Jackson (ice hockey)`. (b) Seed node `Jim Jackson (sportscaster)`.

Figure 6.1: The number of correctly detected elements by MC End Point for seed nodes with the same name.



(a) MC End Point (b) MC Complete Path

Figure 6.2: The number of correctly detected elements for seed node Michael Jackson.



(a) Seed node Michael Jackson Web page. (b) Seed node Jim Jackson Web page.

Figure 6.3: The number of correctly detected elements by MC End Point for two Web pages.

6.3 Variance based performance comparison and CLT approximations

In the MC End Point algorithm the distribution of end points is multinomial [51]. Namely, if we denote by L_j the number of paths that end at node j after m runs, then we have

$$P\{L_1 = l_1, L_2 = l_2, \ldots, L_n = l_n\} = \frac{m!}{l_1! l_2! \cdots l_n!} \pi_1^{l_1} \pi_2^{l_2} \cdots \pi_n^{l_n}. \tag{6.3}$$

Thus, the standard deviation of the MC End Point estimator for the kth element is given by

$$\sigma(\hat{\pi}_k) = \sigma(L_k/m) = \frac{1}{\sqrt{m}} \sqrt{\pi_k(1-\pi_k)}. \tag{6.4}$$

An expression for the standard deviation of the MC Complete Path is more complicated. Define the matrix $Z = (z_{ij}) = [I - \alpha P]^{-1}$ and let N_j be the number of visits to node j by the random walk with the run time geometrically distributed with parameter α. Further, denote by $E_i(\cdot)$ a conditional expectation provided that the random walk starts at $i = 1, \ldots, n$. From (6.2), it follows that

$$\sigma(\hat{\pi}_k) = \frac{(1-\alpha)}{\sqrt{m}} \sigma(N_k) = \frac{(1-\alpha)}{\sqrt{m}} \sqrt{E_s\{N_k^2\} - E_i\{N_k\}^2}. \tag{6.5}$$

First, we recall that

$$E_s\{N_k\} = z_{sk} = \pi_k(s)/(1-\alpha). \tag{6.6}$$

Then, from [52], it is known that $E_s\{N_k^2\} = [Z(2Z_{dg} - I)]_{sk}$, where Z_{dg} is a diagonal matrix having as its diagonal the diagonal of matrix Z and $[A]_{ik}$ is the $(i,k)^{\text{th}}$ element of matrix A. Thus, we write

$$\begin{aligned} E_s\{N_k^2\} &= \mathbf{1}_s^T Z(2Z_{dg} - I)\mathbf{1}_k = \frac{1}{1-\alpha} \pi(s)(2Z_{dg} - I)\mathbf{1}_k \\ &= \frac{1}{1-\alpha} \left(\frac{1}{1-\alpha} \pi_k(s)\pi_k(k) - \pi_k(s) \right). \end{aligned} \tag{6.7}$$

Substituting (6.6) and (6.7) into (6.5), we obtain

$$\sigma(\hat{\pi}_k) = \frac{1}{\sqrt{m}} \sqrt{\pi_k(s)(2\pi_k(k) - (1-\alpha) - \pi_k(s))}. \tag{6.8}$$

Since $\pi_k(k) \approx 1 - \alpha$, we can approximate $\sigma(\hat{\pi}_k)$ with

$$\sigma(\hat{\pi}_k) \approx \frac{1}{\sqrt{m}} \sqrt{\pi_k(s)((1-\alpha) - \pi_k(s))}.$$

Comparing the latter expression with (6.4), we see that MC End Point requires approximately $1/(1-\alpha)$ walks more than MC Complete Path. This was expected as MC End Point uses only

information from end points of the random walks. We would like to emphasize that $1/(1-\alpha)$ can be a significant coefficient. For instance, if $\alpha = 0.85$, then $1/(1-\alpha) \approx 6.7$.

Now, for the MC End Point we can use CLT-type result given e.g. in [76]:

Theorem 6.1 *[76] For large m and $\sum_{i=1}^{n} l_i = m$, a multivariate normal density approximation to the multinomial distribution (6.3) is given by*

$$f(l_1, l_2, \ldots, l_n) = \left(\frac{1}{2\pi m}\right)^{(n-1)/2} \times \left(\frac{1}{n\pi_1\pi_2\cdots\pi_n}\right)^{1/2} \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\frac{(l_i - m\pi_i)^2}{m\pi_i}\right\}. \quad (6.9)$$

For the MC Complete Path, we note that $N(s,r) = (N_1(s,r), \ldots, N_n(s,r))$, $r = 1, 2, \ldots$, form a sequence of i.i.d. random vectors. Hence, we can apply the multivariate central limit theorem. Denote

$$\hat{N}(s,m) = \frac{1}{m}\sum_{r=1}^{m} N(s,r). \quad (6.10)$$

Theorem 6.2 *Let m go to infinity. Then, we have the following convergence in distribution to a multivariate normal distribution*

$$\sqrt{m}\left(\hat{N}(s,m) - \bar{N}\right) \xrightarrow{D} \mathcal{N}(0, \Sigma(s)),$$

where $\bar{N}(s) = \mathbf{1}_s^T Z$ and $\Sigma(s) = E\{N^T(s,r)N(s,r)\} - \bar{N}^T(s)\bar{N}(s)$ is a covariance matrix, which can be expressed as

$$\Sigma(s) = \Omega(s)\, Z + Z^T \Omega(s) - \Omega(s) - Z^T \mathbf{1}_s \mathbf{1}_s^T Z. \quad (6.11)$$

where the matrix $\Omega(s) = \{\omega_{jk}(s)\}$ is defined by

$$\omega_{jk}(s) = \begin{cases} z_{sj}, & \text{if } j = k, \\ 0, & \text{otherwise.} \end{cases}$$

Proof The convergence follows from the standard multivariate central limit theorem. We only need to establish the formula for the covariance matrix. The covariance matrix can be expressed as follows [69]:

$$\Sigma(s) = \sum_{j=1}^{n} z_{sj}\left(\Delta(j)Z + Z\Delta(j) - \Delta(j)\right) - Z^T \mathbf{1}_s \mathbf{1}_s^T Z, \quad (6.12)$$

where $\Delta(j)$ is defined by

$$\delta_{kl}(j) = \begin{cases} 1, & \text{if } k = l = j, \\ 0, & \text{otherwise.} \end{cases}$$

Let us consider $\sum_{j=1}^{n} z_{sj}\Delta(j)Z$ in component form.

$$\sum_{j=1}^{n} z_{sj} \sum_{\varphi=1}^{n} \delta_{l\varphi}(j) z_{\varphi k} = \sum_{j=1}^{n} z_{sj}\delta_{lj} z_{jk} = z_{sl} z_{lk} = \sum_{j=1}^{n} \omega_{lj}(s) z_{jk},$$

and it implies that $\sum_{j=1}^{n} z_{sj}\Delta(j)Z = \Omega(s)Z$. Symmetrically, $\sum_{j=1}^{n} z_{sj}Z\Delta(j) = Z^T\Omega(s)$. Equality $\sum_{j=1}^{n} z_{sj}\Delta(j) = \Omega(s)$ can be easily established. This completes the proof. ∎

We would like to note that in both cases we obtain the convergence to rank deficient (singular) multivariate normal distributions.

Illustrating example with the Web (cont.): In Table 1 we provide means and standard deviations for the number of hits of MC End Point for top-10 nodes with the Jim Jackson Web page as the seed node for the number of runs $m = 10^4$ and $m = 10^5$. We observe that the means are very close to each other and the standard deviations are significant with respect to the values of the means. This shows that a direct application of the central limit theorem and the confidence intervals technique will lead to inadequate stopping criteria. In the ensuing sections we discuss metrics and stopping criteria which are much more efficient for the present problem.

Table 6.1: MC End Point for the Jim Jackson Web page: means and Standard Deviations.

nr. runs	10000		100000	
rank	mean	std	mean	std
1	0.79104	0.012531	0.79748	0.004834
2	0.006329	0.001622	0.006202	0.000569
3	0.005766	0.001075	0.005885	0.000354
4	0.006365	0.002103	0.006561	0.000704
5	0.005183	0.001664	0.005518	0.00055
6	0.005541	0.003766	0.005801	0.001257
7	0.007617	0.003633	0.006243	0.001266
8	0.007566	0.012384	0.005854	0.003969
9	0.006186	0.001468	0.006182	0.000547
10	0.00672	0.003492	0.006223	0.001132

6.4 Convergence based on order

For the two introduced Monte Carlo methods we aim to calculate or estimate a probability that after a given number of steps we correctly obtain top-k list or top-k basket. These are the probabilities $P\{L_1 > \cdots > L_k > L_j, \forall j > k\}$ and $P\{L_i > L_j, \forall i, j : i \leq k < j\}$ respectively, where L_k, $k \in 1, \ldots, n$, can be either the Monte Carlo estimates or the ranked elements or their CLT approximations. We refer to these probabilities as the ranking probabilities and we refer to complementary probabilities as misranking probabilities [21]. Because of combinatorial explosion, exact calculation of these probabilities is infeasible in non-trivial cases. Thus, we propose estimation methods based on Bonferroni inequality. This approach works for reasonably large values of m.

Drawing correctly the top-k basket is defined by the event $\bigcap_{i \leq k < j} \{L_i > L_j\}$. Applying the Bonferroni inequality $P\{\bigcap_s A_s\} \geq 1 - \sum_s P\left\{\bar{A}_s\right\}$ to this event, we obtain $P\left\{\bigcap_{i \leq k < j} \{L_i > L_j\}\right\} \geq 1 - \sum_{i \leq k < j} P\left\{\overline{\{L_i > L_j\}}\right\}$. Equivalently, we can write the following upper bound for the misranking probability

$$1 - P\left\{\bigcap_{i \leq k < j} \{L_i > L_j\}\right\} \leq \sum_{i \leq k < j} P\{L_i \leq L_j\}. \tag{6.13}$$

We note that the upper bound for the misranking probability is very useful, because it will provide a guarantee on the performance of our algorithms. Since in the MC End Point method the distribution of end points is multinomial (see (6.3)), for small m we can directly use the formula

$$P\{L_i \leq L_j\} = \sum_{l_i + l_j \leq m,\ l_i \leq l_j} \frac{m!}{l_i! l_j! (m - l_i - l_j)!} \pi_i^{l_i} \pi_j^{l_j} (1 - \pi_i - \pi_j)^{m - l_i - l_j}. \tag{6.14}$$

For large m it is computationally intractable. Hence, we now turn to the CLT approximations for the both MC methods. Denote by L_j the original number of hits at node j and by Y_j its CLT approximation. First, we obtain a CLT based expression for the misranking probability for two nodes $P\{Y_i \leq Y_j\}$. Since the event $\{Y_i \leq Y_j\}$ coincides with the event $\{Y_i - Y_j \leq 0\}$ and a difference of two normal random variables is again a normal random variable, we obtain $P\{Y_i \leq Y_j\} = P\{Y_i - Y_j \leq 0\} = 1 - \Phi(\sqrt{m}\rho_{ij})$, where $\Phi(\cdot)$ is the cumulative distribution function for the standard normal random variable and

$$\rho_{ij} = \frac{E[Y_i] - E[Y_j]}{\sqrt{\sigma^2(Y_i) - 2\mathrm{cov}(Y_i, Y_j) + \sigma^2(Y_j)}}.$$

For large m, the above expression can be bounded by $P\{Y_i \leq Y_j\} \leq \frac{1}{\sqrt{2\pi}} e^{-\frac{\rho_{ij}^2}{2} m}$. Since the

misranking probability for two nodes $P\{Y_i \leq Y_j\}$ decreases when j increases, we can write

$$1 - P\left\{\bigcap_{i\leq k<j}\{Y_i > Y_j\}\right\} \leq \sum_{i=1}^{k}\left(\sum_{j=k+1}^{j^*} P\{Y_i \leq Y_j\} + \sum_{j=j^*+1}^{n} P\{Y_i \leq Y_{j^*}\}\right),$$

for some j^*. This gives the following upper bound

$$1 - P\left\{\bigcap_{i\leq k<j}\{Y_i > Y_j\}\right\} \leq \sum_{i=1}^{k}\sum_{j=k+1}^{j^*}(1 - \Phi(\sqrt{m}\rho_{ij})) + \frac{n-j^*}{\sqrt{2\pi}}\sum_{i=1}^{k} e^{-\frac{\rho_{ij^*}^2}{2}m}. \tag{6.15}$$

Since we have a finite number of terms in the right hand side of expression (6.15), we conclude that

Theorem 6.3 *The misranking probability of the top-*k *basket goes to zero with geometric rate,* $1 - P\left\{\bigcap_{i\leq k<j}\{Y_i > Y_j\}\right\} \leq Ca^m$, *for some* $C > 0$, $a \in (0,1)$.

We note that for the multinomial distribution, ρ_{ij} has a simple expression

$$\rho_{ij} = \frac{\pi_i - \pi_j}{\sqrt{\pi_i(1-\pi_i) + 2\pi_i\pi_j + \pi_j(1-\pi_j)}}.$$

For MC Complete Path $\sigma^2(Y_i) = \Sigma_{ii}(s)$ and $\mathrm{cov}(Y_i, Y_j) = \Sigma_{ij}(s)$ where $\Sigma_{ii}(s)$ and $\Sigma_{ij}(s)$ can be calculated by (6.11). Similarly the Bonferroni inequality can be applied to the top-k list (see [15]).

6.5 Solution relaxation

In this section we analytically evaluate the relation between the number of experiments m and the average number of correctly identified top-k nodes. We use the relaxation by allowing the latter number to be smaller than k. We aim to mathematically justify the observed "80/20 behavior" of the algorithm: 80 percent of the top-k nodes are identified correctly in a very short time.

Let M_0 be a number of correctly identified elements in the top-k basket. In addition, denote by K_i the number of nodes ranked not lower than i. Formally, $K_i = \sum_{j \neq i} 1\{L_j \geq L_i\}, \quad i = 1, \ldots, k$, where $1\{\cdot\}$ is an indicator function. Placing node i in the top-k basket is equivalent to the event $\{K_i < k\}$, and thus $E(M_0) = E\left(\sum_{i=1}^{k} 1\{K_i < k\}\right) = \sum_{i=1}^{k} P(K_i < k)$. Direct evaluation of $P(K_i < k)$ is computationally intractable in realistic scenarios, even with Markov chain representation techniques [36]. Thus, we use *approximation* and *Poissonisation*.

The End Point algorithm is merely an occupancy scheme where each independent experiment (random walk) results in placing one ball (visit) to an urn (node of the graph). Under Poissonisation [42], we assume that the number of random walks is a Poisson random variable M with given mean m. Because the number of hits in the Poissonised model is different from the number of original hits, we use the notation Y_i instead of L_j for the number of visits to page j. Note that Y_j is a Poisson random variable with parameter $m\pi_j$ and is independent of Y_i for $i \neq j$. The imposed independence of Y_j's greatly simplifies the analysis.

Next to Poissonisation, we also apply *approximation* of M_0 by a closely related measure M_1: $M_1 = k - \sum_{i=1}^{k}(K_i'/k)$, where K_i' denotes the number of pages outside the top-k list that are ranked higher than node $i = 1, \ldots, k$. Note that K_i' is the number of mistakes with respect to node i that lead to errors in the identified top-k list. Then the sum in the definition of M_1 is simply the average number of such mistakes with respect to each of the top-k nodes.

The measure M_1 is more tractable than M_0 because its average value $E(M_1) = k - \frac{1}{k}\sum_{i=1}^{k} E(K_i')$ involves only the average values of K_i' and not their distributions, and because K_i' depends only on the nodes outside the top-k list. Then, we can make use of the following convenient measure $\mu(y)$:

$$\mu(y) := E(K_i'|Y_i = y) = \sum_{j=k+1}^{n} P(Y_j \geq y), \; i = 1, \ldots, k,$$

which implies $E(K_i') = \sum_{y=0}^{\infty} P(Y_i = y)\mu(y), \; i = 1, \ldots, k$. Therefore, we obtain the following expression for $E(M_1)$:

$$E(M_1) = k - \frac{1}{k}\sum_{y=0}^{\infty} \mu(y) \sum_{i=1}^{k} P(Y_i = y). \tag{6.16}$$

Illustrating example with Wikipedia (cont.): Let us calculate $E(M_1)$ for the top-10 basket

corresponding to the seed node `Jim Jackson (ice hockey)`. Using formula (6.16), for $m = 8\times10^3; 10\times10^3; 15\times10^3$ we obtain $E(M_1) = 7.75; 9.36; 9.53$. It took 2000 runs to move from $E(M_1) = 7.75$ to $E(M_1) = 9.36$, but then 5000 runs is needed to advance from $E(M_1) = 9.36$ to $E(M_1) = 9.53$. We see that we obtain quickly 2-relaxation or 1-relaxation of the top-10 basket but then we need to spend a significant amount of effort to get the complete basket. This is indeed in agreement with the Monte Carlo runs (see e.g., Figure 6.1). In the next theorem we explain this "80/20 behavior" and provide indication for the choice of m.

Theorem 6.4 *In the Poisonized End Point Monte Carlo algorithm, if all top-*k *nodes receive at least* $y = ma > 1$ *visits and* $\pi_{k+1} = (1-\varepsilon)a$, $\varepsilon > 1/y$, *then*
(i) to satisfy $E(M_1) > (1-\beta)k$ *it is sufficient to have*

$$\sum_{j=k+1}^{n} \frac{(m\pi_j)^y}{y!}\, e^{-m\pi_j} \left[1 + \sum_{l=1}^{\infty} \frac{(m\pi_j)^l}{(y+1)\cdots(y+l)}\right] < \beta k.$$

(ii) Statement (i) is always satisfied if $m > 2a^{-1}\varepsilon^{-2}[-\log(\varepsilon\pi_{k+1}\beta k)]$.

Proof (i) By definition of M_1, to ensure that $E(M_1) \le (1-\beta)k$ it is sufficient that $E(K_i'|Y_i) \le \beta k$ for each $Y_i \ge y$ and each $i = 1,\dots,k$. Now, (i) follows directly since for each $Y_i \ge y$ we have $E(K_i'|Y_i) \le \mu(y)$ and by definition of $\mu(y)$ under Poissonisation we have

$$\mu(y) = \sum_{j=k+1}^{n} \frac{(m\pi_j)^y}{y!}\, e^{-m\pi_j} \left[1 + \sum_{l=1}^{\infty} \frac{(m\pi_j)^l}{(y+1)\cdots(y+l)}\right]. \tag{6.17}$$

To prove (ii), using (6.17) and the conditions of the theorem, we obtain:

$$\begin{aligned}
\mu(y) &\le \sum_{j=k+1}^{n} \frac{(m\pi_j)^y}{y!}\, e^{-m\pi_j} \left[1 + (1-\varepsilon) + (1-\varepsilon)^2 + \cdots\right] \\
&= \frac{1}{\varepsilon} \sum_{j=k+1}^{n} \frac{(m\pi_j)^y}{y!}\, e^{-m\pi_j} = \frac{1}{\varepsilon} \sum_{j=k+1}^{n} \pi_j \frac{m^y \pi_j^{y-1}}{y!}\, e^{-m\pi_j} \\
&\overset{\{1\}}{\le} \frac{1}{\varepsilon} \frac{m^y \pi_{k+1}^{y-1}}{y!}\, e^{-m\pi_{k+1}} \overset{\{2\}}{\le} \frac{1}{\varepsilon} \frac{1}{\pi_{k+1}} \left(\frac{m\pi_{k+1}}{y}\right)^y e^{y-m\pi_{k+1}} \\
&= \frac{1}{\varepsilon\pi_{k+1}} \left[(1-\varepsilon)e^{\varepsilon}\right]^{ma}.
\end{aligned} \tag{6.18}$$

Here $\{1\}$ holds because $\sum_{j\ge k+1} \pi_j \le 1$ and $(m\pi_j)^{y-1}/(y-1)!\exp\{-m\pi_j\}$ is maximal at $j = k+1$. The latter follows from the conditions of the theorem: $m\pi_{k+1} = (1-\varepsilon)y \le y-1$ when $\varepsilon > 1/y$. In $\{2\}$ we use that $y! \ge y^y/e^y$.

Now, we want the last expression in (6.18) to be smaller than βk. Solving for m, we get:

$$ma(\log(1-\varepsilon) + \varepsilon) < \log(\varepsilon\pi_{k+1}\beta k).$$

Note that the expression under the logarithm on the right-hand side is always smaller than 1 since $\beta < 1$, $\varepsilon < 1$ and $k\pi_{k+1} < 1$. Using $(\log(1-\varepsilon)+\varepsilon) = -\sum_{k=2}^{\infty} \varepsilon^k/k \geq -\varepsilon^2/2$, we obtain (ii). ∎

From (i) we can already see that the 80/20 behavior of $E(M_1)$ (and, respectively, $E(M_0)$) can be explained mainly by the fact that $\mu(y)$ drops drastically with y because the Poisson probabilities decrease faster than exponentially.

The bound in (ii) shows that m should be roughly of the order $1/\pi_k$. The term ε^{-2} is not defining since ε does not need to be small. For instance, by choosing $\varepsilon = 1/2$ we can filter out the nodes with PPR not higher than $\pi_k/2$. This often may be sufficient in applications. Obviously, the logarithmic term is of a smaller order of magnitude.

We note that the bound in (ii) is quite rough because in its derivation we replaced π_j, $j > k$, by their maximum value π_{k+1}. In realistic examples, m can be chosen much smaller than in (ii) of Theorem 6.4. In fact, in our examples good top-k baskets are obtained if the algorithm is terminated at the point when for some y, each node in the current top-k basket has received at least y visits while the rest of the nodes have received at most $y - d$ visits, where d is a small number, say $d = 2$. Such choice of m satisfies (i) with reasonably small α. Without a formal justification, this stopping rule can be understood since we have $m\pi_{k+1} = ma(1-\varepsilon) \approx ma - d$, which results in a small value of $\mu(y)$.

6.6 Conclusions

We have analyzed two Monte Carlo type methods (MC End Point and MC Complete Path) for quick detection of top-k lists or baskets of nodes with largest values of Personalized PageRank. Both methods have light complexity and can be easily implemented in a parallel fashion. We have demonstrated that MC Complete Path is the superior method with respect to MC End Point. Our analysis emphasizes that in this setting it is better to deal with order than with the numerical values. Finally, we have shown that allowing a couple of mistakes in the top-k list or basket significantly reduces computation time.

7

ALPHA CURRENT FLOW CENTRALITY

7.1 Introduction and summary of the results

A class of centrality measures called betweenness centralities reflects degree of participation of edges or nodes in communication between different parts of the network. The first notion of betweenness centrality was introduced by Freeman [41]. Let $s, t \in V$ be a pair of nodes in an undirected graph $G = (V, E)$. (In the present chapter we restrict our consideration to undirected graphs.) We denote $|V| = n$, $|E| = m$, and let d_v be the degree of node v. Let $\sigma_{s,t}$ be the number of shortest paths connecting nodes s and t and denote by $\sigma_{s,t}(e)$ the number of shortest paths connecting nodes s and t passing through edge e. Then betweenness centrality of edge e is calculated as follows:

$$C_B(e) = \frac{1}{n(n-1)} \sum_{s,t \in V} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}}. \tag{7.1}$$

Computational complexity of the best known algorithm for computing the betweenness in (7.1) is $\mathcal{O}(mn)$ [28]. This limits its applicability for large graphs.

One shortcoming of the betweenness centrality in (7.1) is that it takes into accounts only the shortest paths, ignoring the paths that might be one or two steps longer, while the edges on such paths can be important for communication processes in the network. In order to take such paths into account, Newman [70] and Brandes and Fleischer [29] introduced the current flow betweenness centrality (CF-betweenness). In [70, 29] the graph is regarded as an electrical network with edges being unit resistances. The CF-betweenness of an edge is the amount of current that flows through it, averaged over all source-destination pairs, when one unit of current is induced at the source, and the destination (sink) is connected to the ground. This

exploits the well known relation between electrical networks and reversible Markov chains, see e.g., [5, 38].

The computational difficulty of Betweenness and the CF-betweenness is that the computations must be done over the set of all source-destination pairs. The best previously known computational complexity for the CF-betweenness is $\mathcal{O}(I(n-1)+mn\log n)$ where $I(n-1)$ is the complexity of the inversion of matrix of dimension $n-1$.

In the present chapter we introduce and analyse new betweenness centrality measures: α-current flow betweenness (α-CF betweenness) and its truncated version. The main purpose of these new measures is to bring down the high cost of the CF-betweenness computation. Our proposed measures are very close in performance to the CF-betweenness, but they are comparable to the PageRank algorithm [31] in their modest computational complexity. Our goal is to provide and analyze efficient algorithms for computing α-CF betweenness and truncated α-CF betweenness, and to compare the α-CF betweenness to other centrality measures.

As we shall see, the new centrality measures are particularly well suited for detection of network vulnerability. The betweenness centrality measures can also be used for classification purposes [63].

This chapter is mainly based on the article:
K. Avrachenkov, N. Litvak, V. Medyanikov and M. Sokol, "Alpha current flow betweenness centrality", In the 10th International Workshop on Algorithms and Models for the Web Graph, WAW 2013.

7.2 Alpha current flow betweenness

We view the graph G as an electrical network where each edge has resistance α^{-1}, and each node v is connected to the ground node, node $n+1$, by an edge with resistance $(1-\alpha)^{-1}d_v^{-1}$. This is in the spirit of PageRank. Indeed, the current (probability flow) is inversely proportional to the resistance. Thus, the fraction α of the current from node v flows to the network, while fraction $(1-\alpha)$ of the current is directed to the sink. Since the graph is undirected, we use a convention that (v,w) and (w,v) represent the same arc in E, but depending on the chosen direction the current along this arc is considered to be positive or negative.

Assume that a unit of current is supplied to a source node $s \in V$, and there is a destination node $t \in V$ connected to the ground. Let $\varphi_v^{(s,t)}$ denote the absolute potential of node $v \in V$, if s is the source and t is the destination. Assume without loss of generality that $s = 1$ and $t = n$ ($\varphi_n^{(1,n)} = \varphi_{n+1}^{(1,n)} = 0$, i.e., we set the ground potential to zero). The vector of absolute potentials of the other nodes $\varphi^{(1,n)} = [\varphi_1^{(1,n)}, ..., \varphi_{n-1}^{(1,n)}]^T$ is a solution of the following system of equations (Kirchhoff's current law):

$$[\tilde{D} - \alpha\tilde{A}]\varphi^{(1,n)} = \tilde{b}, \tag{7.2}$$

where $\tilde{D}$ and $\tilde{A}$ are the degree and adjacency matrices of the graph without node n and $\tilde{b} = [1, 0, ..., 0]^T$, see e.g., [29].

In the following theorem we demonstrate that we do not need to solve a separate linear system for each source-destination pair, it suffies to invert the coefficient matrix $[D - \alpha A]$.

Theorem 7.1 *The voltage drop along the edge (v,w) is given by*

$$\varphi_v^{(s,t)} - \varphi_w^{(s,t)} = (c_{s,v} - c_{s,w}) + \frac{c_{s,t}}{c_{t,t}}(c_{t,w} - c_{t,v}), \tag{7.3}$$

where $(c_{v,w})_{v,w\in V}$, are the elements of the matrix $C = [D - \alpha A]^{-1}$.

Proof: Assume again without loss of generality that $s = 1$ and $t = n$. The matrix $[D - \alpha A]$ can be written in the following block form

$$D - \alpha A = \begin{bmatrix} \tilde{D} - \alpha\tilde{A} & -\alpha\tilde{a} \\ -\alpha\tilde{a}^T & d_n \end{bmatrix}, \quad \text{with} \quad \tilde{a} = [a_{1,n}, a_{2,n}, \ldots, a_{n-1,n}]^T.$$

Then, divide accordingly the elements of the inverse matrix

$$C = [D - \alpha A]^{-1} = \begin{bmatrix} \tilde{C} & \tilde{c} \\ \tilde{c}^T & c_{n,n} \end{bmatrix}.$$

Writing the relation $[D - \alpha A]C = I$ in the block form yields

$$[\tilde{D} - \alpha\tilde{A}]\tilde{C} - \alpha\tilde{a}\tilde{c}^T = I, \tag{7.4}$$

$$[\tilde{D} - \alpha\tilde{A}]\tilde{c} - \alpha\tilde{a}\tilde{c}_{n,n} = 0. \tag{7.5}$$

Premultiplying equation (7.4) by $[\tilde{D} - \alpha\tilde{A}]^{-1}$, we obtain

$$[\tilde{D} - \alpha\tilde{A}]^{-1} = \tilde{C} - \alpha[\tilde{D} - \alpha\tilde{A}]^{-1}\tilde{a}\tilde{c}^{T}. \tag{7.6}$$

And premultiplying (7.5) by $[\tilde{D} - \alpha\tilde{A}]^{-1}$, we obtain

$$\alpha[\tilde{D} - \alpha\tilde{A}]^{-1}\tilde{a} = \frac{1}{c_{n,n}}\tilde{c}. \tag{7.7}$$

Combining both equations (7.6) and (7.7) gives

$$[\tilde{D} - \alpha\tilde{A}]^{-1} = \tilde{C} - \frac{1}{c_{n,n}}\tilde{c}\tilde{c}^{T},$$

and hence $\varphi^{(1,n)} = [\tilde{D} - \alpha\tilde{A}]^{-1}\tilde{b} = \tilde{C}_{*,1} - \frac{c_{1,n}}{c_{n,n}}\tilde{c}$. Thus, we can write

$$\varphi_v^{(1,n)} - \varphi_w^{(1,n)} = (c_{v,1} - c_{w,1}) + \frac{c_{1,n}}{c_{n,n}}(c_{w,n} - c_{v,n}).$$

The above expression is symmetric and can be rewritten for any source-target pair (s,t). That is,

$$\varphi_v^{(s,t)} - \varphi_w^{(s,t)} = (c_{v,s} - c_{w,s}) + \frac{c_{s,t}}{c_{t,t}}(c_{w,t} - c_{v,t}).$$

Furthermore, since matrix C is symmetric for undirected graphs, we can rewrite the above equation as

$$\varphi_v^{(s,t)} - \varphi_w^{(s,t)} = (c_{s,v} - c_{s,w}) + \frac{c_{s,t}}{c_{t,t}}(c_{t,w} - c_{t,v}),$$

which completes the proof. □

The potentials $\varphi_v^{(s,t)}$, $v, s, t \in V$, have a clear probabilistic interpretation. Take again $s = 1$ and $t = n$. Then from (7.2) we readily obtain

$$\varphi_v^{(1,n)} = \mathbf{e}_v^{T}[\tilde{D} - \alpha\tilde{A}]^{-1}\tilde{b} = \mathbf{e}_v^{T}[I - \alpha\tilde{P}]^{-1}\tilde{D}^{-1}\tilde{b}, \tag{7.8}$$

where $\mathbf{e}_v$ is a v-th standard basis column vector, and $\tilde{P}$ is the transition probability matrix for a simple random walk on G with absorption in n. Compare this to the well-known expression for the Personalized PageRank vector $\pi(v) = (\pi_1(v), \ldots, \pi_n(v))$ with teleportation preference concentrated in v and damping factor α: $\pi(v) = (1-\alpha)\mathbf{e}_v^{T}[I - \alpha P]^{-1}$. Note that the vector $\tilde{\pi}(v) = (1-\alpha)\mathbf{e}_v^{T}[I - \alpha\tilde{P}]^{-1}$ is very similar to $\pi(v)$, except it nullifies the contribution of node n. Now, recall that $\tilde{b} = (1, 0, \ldots, 0)^{T}$ to obtain

$$\varphi_v^{(1,n)} = (1-\alpha)^{-1}\tilde{\pi}_1(v)d_1^{-1}.$$

Furthermore, let $\mathbf{1}$ be a column vector of ones. Recall that the PageRank vector with uniform teleportation can be written as $\pi = \frac{1-\alpha}{n}\mathbf{1}^T[I - \alpha P]^{-1}$, and define a similar vector $\tilde{\pi} = \frac{1-\alpha}{n}\mathbf{1}^T[I - \alpha\tilde{P}]^{-1}$. Then

$$\sum_{v\in V} \varphi_v^{(1,n)} = n(1-\alpha)^{-1}\tilde{\pi}_1 d_1^{-1}.$$

It is well-known (see e.g., [43] and references therein) and is also confirmed by our experiments that the PageRank of a node in an undirected graph is strongly correlated to the degree of the node. Thus, with any choice of the source, the sum of the potentials is of similar magnitude, except for the cases when the destination node has a large contribution into the PageRank mass of the source.

Finally, we note that the source node has the highest potential, and from [5, Chapter 3, Section 3] we find

$$\begin{aligned}\varphi_1^{(1,n)} &= [P(\text{random walk returns to node 1 before absorption})]^{-1} \\ &= E(\#\text{ returns to node 1 before absorption}).\end{aligned}$$

Now we are ready to define α-CF betweenness. The current $I_e^{(s,t)}$ through edge $e = (v, w)$ is equal to $\alpha(\varphi_v^{(s,t)} - \varphi_w^{(s,t)})$. Let

$$x_e^{(s,t)} = |\varphi_v^{(s,t)} - \varphi_w^{(s,t)}|, \quad (v, w) \in E,$$

be the difference of potentials, that determines the absolute value of the current on the edge. The α-CF betweenness of edge e is defined by

$$x_e^{\alpha} = \frac{1}{n(n-1)} \sum_{s,t\in V, s\neq t} x_e^{(s,t)}, \quad e \in E. \tag{7.9}$$

Further, for each node $v \in V$ its α-CF betweenness is defined as the sum of the α-CF betweenness scores of its adjacent edges:

$$\alpha\text{-CF betweenness}(v) = \sum_{(v,w)\in E} x_{(v,w)}^{\alpha}, \quad v \in V. \tag{7.10}$$

With this definition, the node is central if a relatively large amount of current flows from this node to the network. This is in accordance to the original CF-betweenness of [29, 70], except we introduced the additional sink ground node $n + 1$. This mitigates the computational issues because the original CF-betweenness requires the inversion of the ill-conditioned matrix $[\tilde{D} - \tilde{A}]$, while for computing α-CF betweenness we need to invert the matrix $[D - \alpha A]$, which is a well posed problem, and has many efficient solutions (e.g., power iteration and Monte Carlo methods). In fact, as we shall show below, we need to obtain just a few rows of the inverse matrix $[D - \alpha A]^{-1}$. In the rest of the chapter we will discuss the computation and the properties of the α-CF betweenness.

7.3 Computation of α-CF betweenness

Due to the presence of the auxiliary node $n+1$, the value of $x_e^{(s,t)}$ on the right-hand side of (7.9) can be computed efficiently with high precision for any source-destination pair. However, the summation over all $n(n-1)$ pairs is a problem of prohibitive computational complexity even for graphs of modest size. The solution is to perform the computations for sufficiently many source-destination pairs. Since all source-destination pairs contribute equally in (7.9) we choose to sample them uniformly at random. This results in the next algorithm for computing the α-CF betweenness.

Algorithm 1.

1. Select a set of pairs of nodes $(s_i, t_i), i = 1, ..., N$, uniformly at random;
2. For each s_i and t_i, $i = 1, ..., N$ compute the rows $c_{s_i,*}$, $c_{t_i,*}$ (this can be done either by power iteration or by Monte Carlo algorithm);
3. For each edge $e = (v, w)$ and each pair (s_i, t_i), use (7.3) to compute
$$x_e^{(s_i,t_i)} = |\varphi_v^{(s_i,t_i)} - \varphi_w^{(s_i,t_i)}|.$$
4. Average over source-destination pairs
$$\bar{x}_e^{\alpha} = \frac{1}{N}\sum_{i=1}^{N} x_e^{(s_i,t_i)}.$$

Since we chose the pairs (s_i, t_i) uniformly at random then for every edge e, $\bar{x}_e^{\alpha}$ is just a sample average where all values are between zero and one. Then using the standard approach for the analysis of the series of independent random variables we have the following result.

Theorem 7.2 *Algorithm 1 approximates the alpha current flow betweenness in* $\mathcal{O}(m\log(n)\varepsilon^{-2}\log(\varepsilon)/\log(\alpha))$ *time to within an absolute error of* ε *with arbitrarily high fixed probability.*

Proof: In addition to the proof of Theorem 3 in [29] we just need to note that we can compute Personalized PageRank with precision ε in $\mathcal{O}(\log(\varepsilon)/\log(\alpha))$ power iterations. □

We note that with fixed values of ε and α the computational complexity of Algorithm 1 is $\mathcal{O}(m\log(n))$, which is much better than $\mathcal{O}(mn)$ for the standard betweenness centrality [28] and also than $\mathcal{O}(I(n-1)+mn\log n)$ for the CF-betweenness [29, 70].

7.4 Truncated α-CF betweenness

In the experiments we noticed that the values $x_e^{(s,t)}$ have a high variance, which results in poor precision when evaluating x_e^{α}. A closer analysis revealed that the edges adjacent to the source s receive large values of $x_e^{(s,t)}$, especially when $e = (v, s)$, where v has degree 1, so (v, s) is its only edge, and s has a large degree. This can be explained using the random walk interpretation. Consider a PageRank-type random walk on G. At each node, with probability α, the random walk traverses a randomly chosen edge of this node, and with probability $1 - \alpha$ it jumps to the sink, node $n+1$. Denote by T_B the number of steps of the random walk needed to hit set B. It follows from Proposition 10 of [5, Chapter 3] that $\varphi_v^{(s,t)}/\varphi_s^{(s,t)} = P_v(T_{\{s\}} < T_{\{t,n+1\}})$, where $P_v(\cdot)$ is a conditional probability given that the random walk starts at v. Hence, if s is the only neighbor of v then $\varphi_v^{(s,t)}/\varphi_s^{(s,t)} = \alpha$, the probability of no absorption before reaching s. Thus, $|\varphi_s^{(s,t)} - \varphi_v^{(s,t)}| = (1-\alpha)\varphi_s^{(s,t)}$, which can be large if e.g. $\alpha = 0.8$ because $\varphi_s^{(s,t)}$ is the largest potential in the network. In contrast, the original CF-betweenness corresponds to $\alpha = 1$, implying that the current in (v, s) is zero.

This prompts us to introduce the truncated version of α-CF betweenness where for each edge (v, w) we only take into account the scores $x_{(v,w)}^{(s,t)}$ if $v, w \neq s$. In Fig. 7.1 we present log-linear plots of the empirical complementary distribution function of $x_{(v,w)}^{(s,t)}$ over all pairs (s, t) (solid line), and its truncated version (dashed line). The plots are given for two edges in the Dolphin social network described in Section 7.5 below. Nodes 1 and 36 are central in the network, so the high α-CF betweenness of (1,36) is expected. Node 60 has degree 1, so edge (32,60) gains an unwanted high betweenness in the non-truncated version.

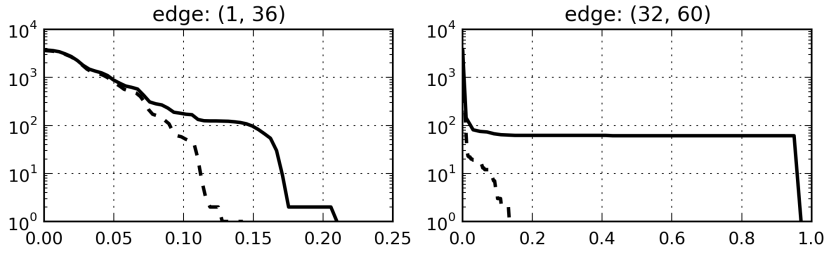


Figure 7.1: The number of pairs s, t with $x_{(v,w)}^{(s,t)} > x$ over all pairs (s, t) (solid line) and only pairs with $v, w \neq s$ (dashed line).

Since the truncated α-CF betweenness gives lower scores to the edges connected to nodes of degree 1, one can expect that it has a higher correlation with CF-betweenness, especially for middle-range values of α. This is confirmed below in Fig. 7.2. Moreover, the truncated version removes outliers, and does not have large spread in values, thus standard statistical procedures, based on the Central Limit Theorem can be applied. Also, because of the smaller

variance, Algorithm 1 achieves a desired precision with a smaller sample of source-destination pairs.

7.5 Datasets

We consider the four graphs described below.

Dolphin social network. This small graph represents a social network of frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand [61].

Graph of VKontakte social network. We have collected data from a popular Russian social network VKontakte. We were considering subgraph representing one of the connected components of people who stated that they were studying at Applied Mathematics - Control Processes Faculty at the St. Petersburg State University in different years. We ran the breadth-first search (BFS) algorithm starting at one specific node of the network and then anonymized the obtained users' data leaving only information about connections between people. Collected network consists of 2092 individuals out of total 8859 denoted the specified faculty in the Education field.

Watts-Strogatz model. As an artificial example, we used a random graph generated by the Watts-Strogatz model. We have chosen this model as it combines high clustering and short average path length, thus different centrality measures give very different results on this graph. For other random models considered (Erdös-Rényi and Barabási-Albert) all measures are highly correlated and behave very similarly to each other.

Enron graph. Enron email communication network is a well known test dataset. It covers all the email communication within a dataset of around half million emails between Enron's employees. The node are e-mail addresses, and an edge appears if an e-mail message was sent from one e-mail to another. Although this graph is small compared to, say, web or Twitter samples, it is already prohibitively large for computing the CF-betweenness in its original form.

	$\|V\|$	$\|E\|$	$\langle deg(v)\rangle$	$diam(G)$	$C_{\text{clustering}}$	$\langle d(u,v)\rangle$
Dolphin social network	62	159	5.13	8	0.259	3.357
VKontakte AMCP social graph	2092	14816	14.16	14	0.338	4.598
Watts-Strogatz	1000	6000	12.00	6	0.422	3.713
$(n = 1000, k = 12, p = 0.150)$						
Enron	36692	183831	10.02	11	0.4970	≈ 4.8

Table 7.1: Datasets characteristics.

7.6 Numerical results for α-CF betweenness

To begin with, we compare the two versions of α-CF betweenness (truncated and without truncation) to the CF-betweenness scores defined as in [29, 70]. Fig. 7.2 presents the results for the three smaller graphs, in which the latter measure could be computed. As a correlation measure we use the Kendall tau rank correlation. We observe that the truncated version is better

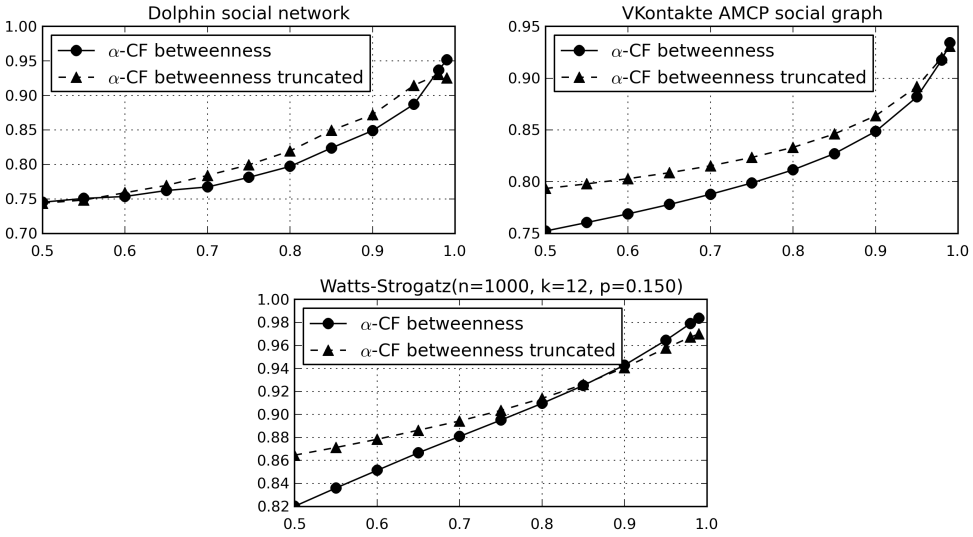
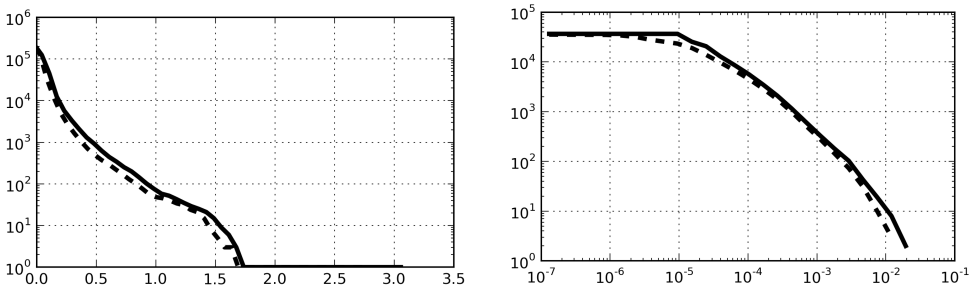


Figure 7.2: Correlations between α-CF betweenness and truncated α-CF betweenness with CF-betweenness as a function of α.

correlated with the CF-betweenness when α is not very close to one. As explained above, this is because the high probability of absorption in the auxiliary node $n+1$ results in a relatively high current in the edges connected to the source, which is not necessarily the case if absorption is only possible in the destination node.

Next, we demonstrate that we can compute α-CF betweenness in the Enron graph, where the computation of CF-betweenness is infeasible (at least, with our means). We have evaluated α-CF betweenness, non-truncated and truncated, with $\alpha = 0.98$. We have run Algorithm 1 using $N = 20 \cdot 10^6$ source-destination pairs. In Fig. 7.3 we plot the complementary distribution function in log-linear scale, of the score $x_e^{0.98}$ across the edges.

Note that distribution over edges (the left plot in Fig. 7.3) does not have a large spread of values, except one outlier edge that connects two most important hubs. Since the weights of the edges are comparable, it is to be expected that in this graph the nodes of large degrees are also the ones with highest betweenness. Indeed, the Kendall's tau correlation between α-CF betweenness and degree of the nodes turns out to be 0.808, which is higher than in the three smaller graphs. The reason can be either the graph size or its structure. In future research we



(a) $10^4 \cdot x_e^{0.98}$ for edges $e \in E$. (b) α-CF betweenness (v) for $v \in V$.

Figure 7.3: Distribution of α-CF betweenness scores in the Enron graph, truncated (dashed line) and not truncated (solid line). On the x-axis are the values of α-CF betweenness, on the y-axis the number of edges/nodes with the score larger than x.

will investigate how the CF-betweenness score, e.g. its maximum value across the edges, scales with the graph size in graphs with power law degrees.

We further present correlations between our proposed measures and other measures of betweenness. These are computed on smaller graphs where we could obtain exact values of all presented measures, see Tables 7.2–7.4. For completeness, we have also included PageRank (PR) computed with $\alpha = 0.85$ and a distance-base centrality measure – Closeness centrality:

$$C_C(v) = \frac{n-1}{\sum_{w \in V, w \neq v} d(v,w)},$$

where $d(v,w)$ is the graph distance between v and w. Betweenness (Between.) is computed as in (7.1).

	Degree	PR	Closeness	Between.	CF	αCF(0.8)	αCF-tr(0.8)	αCF(0.98)
Degree	1.000	0.930	0.548	0.665	0.737	0.864	0.855	0.769
PageRank	0.930	1.000	0.458	0.658	0.733	0.872	0.827	0.757
Closeness	0.548	0.458	1.000	0.578	0.575	0.515	0.573	0.591
Between.	0.665	0.658	0.578	1.000	0.829	0.749	0.759	0.828
CF	0.737	0.733	0.575	0.829	1.000	0.798	0.820	0.939
αCF(0.8)	0.864	0.872	0.515	0.749	0.798	1.000	0.925	0.838
αCF-tr(0.8)	0.855	0.827	0.573	0.759	0.820	0.925	1.000	0.876
αCF(0.98)	0.769	0.757	0.591	0.828	0.939	0.838	0.876	1.000

Table 7.2: Kendall tau for centrality measures in Dolphin social network.

Note that α-CF betweenness is strongly correlated with CF-betweenness. The Closeness Centrality does not agree well with the CF-betweenness, even the PageRank and the degrees

	Degree	PR	Closeness	Between.	CF	αCF(0.8)	αCF-tr(0.8)	αCF(0.98)
Degree	1.000	0.655	0.679	0.521	0.545	0.659	0.668	0.599
PageRank	0.655	1.000	0.375	0.662	0.717	0.833	0.811	0.766
Closeness	0.679	0.375	1.000	0.382	0.356	0.424	0.445	0.395
Between.	0.521	0.662	0.382	1.000	0.761	0.760	0.749	0.778
CF	0.545	0.717	0.356	0.761	1.000	0.812	0.833	0.917
αCF(0.8)	0.659	0.833	0.424	0.760	0.812	1.000	0.938	0.878
αCF-tr(0.8)	0.668	0.811	0.445	0.749	0.833	0.938	1.000	0.903
αCF(0.98)	0.599	0.766	0.395	0.778	0.917	0.878	0.903	1.000

Table 7.3: Kendall tau for centrality measures in the social graph VKontakte AMCP.

	Degree	PR	Closeness	Between.	CF	αCF(0.8)	αCF-tr(0.8)	αCF(0.98)
Degree	1.000	0.891	0.462	0.526	0.610	0.643	0.581	0.612
PageRank	0.891	1.000	0.415	0.485	0.565	0.610	0.546	0.567
Closeness	0.462	0.415	1.000	0.655	0.613	0.647	0.666	0.628
Between.	0.526	0.485	0.655	1.000	0.853	0.819	0.852	0.857
CF	0.610	0.565	0.613	0.853	1.000	0.910	0.914	0.979
αCF(0.8)	0.643	0.610	0.647	0.819	0.910	1.000	0.935	0.923
αCF-tr(0.8)	0.581	0.546	0.666	0.852	0.914	0.935	1.000	0.930
αCF(0.98)	0.612	0.567	0.628	0.857	0.979	0.923	0.930	1.000

Table 7.4: Kendall tau for centrality measures in the Watts-Strogatz graph (n=1000, k=12, p=0.150).

have a higher correlations with the CF-betweenness in real graphs. Recent paper [24] suggests more measures based on distance, and efficient computation methods for such measures is presented in [25]. In the future it will be interesting to compare these new measures to α-CF betweenness.

7.7 Centrality measures and network vulnerability

We now consider how well the CF-betweenness and α-CF betweenness can indicate the nodes responsible for maintaining connectivity of a network. We follow the methodology of [47]. As measures of connectivity we choose the average inverse distance

$$< d^{-1} >= \frac{1}{n(n-1)} \sum_{u,v \in V, u \neq v} \frac{1}{d(u,v)}$$

and the size of the largest connected component. In the experiment, we remove the top nodes one by one, according to different centrality measures, and observe how the connectivity of the network changes. In Fig. 7.4 the results are presented for the inversed average distance.

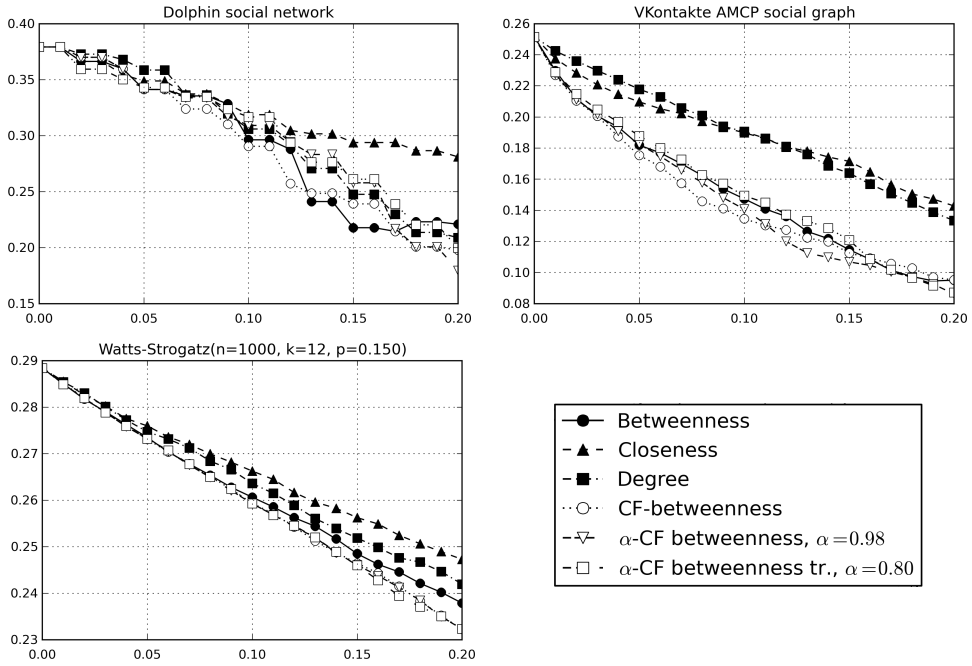


Figure 7.4: Inverse average distance as a function of the fraction of removed top-nodes according to different betweenness centrality measures.

The results for the social graph VKontakte are especially interesting, because this network turns out to be less vulnerable to the removal of nodes with large degree than nodes with large betweenness and its modifications (CF-betweenness, α-CF betweenness, and truncated α-CF betweenness). On the small Dolphin social network there is no much difference in vulnerability with respect to different centrality measures. Finally, on the artificial Watts-Strogatz graph the CF-betweenness and our proposed two versions of α-CF betweenness find the nodes that are most essential for the network connectivity.

In Fig. 7.5 we plot the size of the largest connected components against the fraction of removed top-nodes. We do not present the plot for the Watts-Strogatz graph because it remains entirely connected, so the size of its largest connected component equals to the number of remaining nodes irrespectively of which nodes are removed first. For the two real graphs, the CF-betweenness is most efficient in reducing the size of the giant component. On the Dolphin graph, α-CF betweenness performs closely to CF-betweenness, except the interval when 13-18% of nodes are removed. On the graph VKontakte, α-CF betweenness and its truncated version perform comparably to the CF-betweenness. Again, on this graph, degree and Closeness centrality fail to reveal the nodes responsible for good network connectivity. The α-CF between-

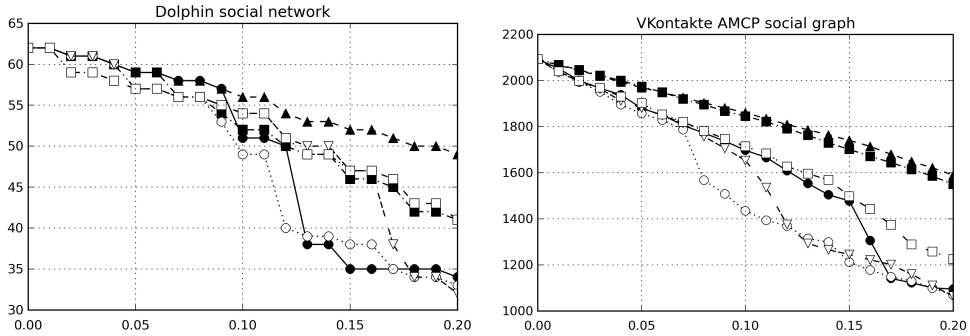


Figure 7.5: The size of the largest connected component as a function of the fraction of removed top-nodes according to different betweenness centrality measures.

ness with $\alpha = 0.98$ appears to be a better indicator for vulnerability than the truncated α-CF betweenness with $\alpha = 0.8$. The latter however also gives good results, and can be computed easier on large graphs due to the faster convergence of the power iteration algorithm.

We conclude that both α-CF betweenness and truncated α-CF betweenness provide an adequate measure for the role of a node in network's connectivity. Furthermore, their computational costs are lower than for known measures of betweenness, and the computations can be done in parallel easily. Thus, α-CF betweenness can be applied in large graphs, for which computation of other measures of betweenness is merely infeasible.

7.8 Conclusions

The original shortest-path betweenness centrality is based on counting shortest paths which go through a node or an edge. One shortcoming of the shortest-path betweenness centrality is that it ignores the paths that might be one or two hops longer than the shortest paths, while the edges on such paths can be important for communication processes in the network. To rectify this shortcoming a current flow betweenness centrality has been proposed. However, similarly to the shortest-path betweenness, it has prohibitive computational complexity for large size networks. We have proposed two regularizations of the current flow betweenness centrality, α-current flow betweenness and truncated α-current flow betweenness, which can be computed fast and correlate well with the original current flow betweenness. We have demonstrated that the new centrality measures can efficiently be applied to study the network vulnerability.

8

APPENDIX: SOFTWARE DESCRIPTION

8.1 Summary

This library helps to apply semi-supervised learning methods [12] and Personalized PageRank method to large graphs. The implementation could work with any kind of graph which could be implemented from our graph interface. We supposed that the graph is without loops and multiedges.

In particular, the library provides:

1. the convenient way to work with graphs in BVGraph[26] and simple readable format
2. the namers to make the nodes with names
3. the semi-supervised learning classification
4. the expert nodes/seed nodes format to load/save them
5. the estimations to get the precision, recall and confusion matrix

8.2 Description

8.2.1 The formats

The library provides the convenient way to work with graphs in BVGraph[26] and simple readable format. Any type of graphs could be used, if they implements our graph interface.

BVGraph format

The load and save operations are very similar as in [26]. The directory *yourpath* contains several files with the same prefix *yourBVGraph*. If exist files with suffix *labels* the weghted graph will be loaded.

```
String fileName = "/yourpath/yourBVGraphPrefix";
IGraph graph = new BVGGraphLoader().load(fileName);
BVGGraphSaver().save(graph, fileName);
```

Matlab format

This is a simple edge list with or without weights. The numeration of the nodes starts from 1. Each line of the file contains "nodeFrom nodeTo weight" for weighted graphs and "nodeFrom nodeTo" for unweighted graphs.

```
String fileName = "/yourpath/yourmatlabfile.txt"
graph = MatlabLoader().load(fileName);
MatLabSaver().save(graph, fileName);
```

8.2.2 The namers

The library provides the simple realization of graph namer. It is a file with a column of names, each line contain one name of the node. The number of line corresponds to node id. Also you can use a simple identical to node id namer. Note:To make it compatible with BVGraph library the names should be sorted in alphabetical order, not necessary for our library.

```
String pathToNamer = "yourGraphNamer.txt";
INodeNamer namer = NodeNamerImpl.load(new File(pathToNamer));
INodeNamer nameIdent = new IdenticalIdNodeNamer();
graph.getProperties().setValue(IGraph.NODE_NAMER, namer);
```

8.2.3 The algorithms

The library provides the implementation of semi-supervised methods in general case with parameter sigma. Also the library provides the implementations of particular algorithms: PageRank , Standard Laplacian, Normalized Laplacian methods, which could be significantly faster for large graphs in BVG format. BVG format compress graph and any operation to access the data could take time.

Mainly use the general implementation to compute methods. If it is slow, then try the particular implementations of the methods.

The personal pagerank calculation (the implementation of particular method)

```
List<Integer> listNodeIds;
IPersonalVector pV = new PersonalVector(listNodeIds,graph.numNodes());
CalculatorInputData setup = new CalculatorInputData(alpha);
ICalculator pc = CalculatorFactory.getPageRankCalculator(graph, setup);
double[] ppr = pc.compute(pV.getPersonalVector());
```

The CalculatorInputData has defaults values as marked in Constants: epsilon 0.000001, max number of iterations 100, alpha 0.5.

The general personal pagerank calculation (based on sigma parameter):

```
CalculatorInputDataSigma setupS = new CalculatorInputDataSigma(sigma, alpha);
//setupS.setWeighted(true);
ICalculator pc = CalculatorFactory.getGeneralCalculator(graph, setupS);
double[] gppr = pc.compute(pV.getPersonalVector());
```

The semi-supervised learning classification takes as input seeds and one of the methods, the output is an array with values corresponds to class id, if value equals to -1, mean that item unclassified.

```
SemiSupervisedLearningMethod sslm = new SemiSupervisedLearningMethod();
int[] nodeToClassId = sslm.run(seedsInput, pc);
```

8.2.4 The expert/seeds files

Provides a convinient way to work with seeds/expert files. The format for seeds/expert files shown on example. The file contains an optional short explanation about classification, the name of the class and the list of node names in it, then empty line and so on.

```
Short explanation about file. Could be empty line.

Class Name A
NodeName1
NodeName2

Class Name B
NodeName3
```

```
IClustering seeds = ClusteringUtils.loadFromFile(new File(seedsName));
ISeedsInput seedsInput =
    ClusteringConverter.clustering2SeedsInput(seeds, namer, graph.numNodes());
IClustering resultClassification =
    ClusteringConverter.array2Custering(nodeToClassId, namer, seeds);
ClusteringUtils.saveToFile(resultClassification, new File("someName.txt"));
```

8.2.5 The estimations

Provides methods to get precision, recall and confusion matrix.

```
IClustering expert; IClustering alg;
ClusteringPreprocessor algClustering = new ClusteringPreprocessor(alg);
PrecisionQuantity prec =
    new PrecisionQuantity(new ClusteringPreprocessor(expert));
Pair<Double, Double> res =  prec.getPrecisionRecall(algClustering);
//res.getFirst() — precision, res.getSecond() — recall
prec.printConfusionMatrix(algClustering, new PrintWriter(System.out))
```

8.2.6 How to write your own graph implementation

The library could work with any kind of graph which could be implemented from our graph interface. *IGraph* is a simple interface, which provides information about number of nodes, node degrees and list of their outgoing nodes, named as the list of successors. For the compatibility with webgraph library the list of successors may contain more entries than the node outdegree, and the list should be sorted in increasing order. The implementation of properties could be taken from the library. Also, the library does not require the correct implementation for the number of arcs.

```
int numNodes();
long numArcs();
int outdegree(int node);
int[] successorArray(int node);
IProperties getProperties();
```

9

Conclusions and Future Research

The thesis consists of two parts. In the first part, we have studied the graph-based semi-supervised learning methods. We have proposed a new generalized optimization framework, which gives as particular cases the Standard Laplacian method, the Normalized Laplacian method and the PageRank based method. There was no optimization formation for the PageRank based method. We have shown that among all methods which can be obtained from our optimization framework, the PageRank method is the only method robust with respect to unbalanced data. Using the theory of random walks on graphs, we have elaborated the following recommendations: if possible, choose labelled points with large degrees. Then, adopt the Standard Laplacian method with α in the upper-middle range of the interval $(0, 1)$. If finding large degree points is not feasible or recall is more important than precision for small classes, choose the PageRank based method. The theory of random walks on graph has also allowed us to demonstrate a decomposition of the classification kernel with respect to the two basic parameters (α and σ). We have also shown that if all the labelled points have the same degree, all the semi-supervised learning methods from the analysed family give the same classification results. In particular, this explains why the classification results of several semi-supervised learning methods are nearly the same when the k-Nearest Neighbour algorithm is used to construct the similarity matrix. All our theoretical results are illustrated by experiments with real and synthetic data. In particular, we have performed extensive experiments with users and content classification in P2P systems. We show that the semi-supervised learning approach gives excellent classification result and the technique scales very well to large volumes of data. Just to give an example, in the dataset of 1 126 670 users, using only 50 labelled points for each language, we are able to classify the users according to their preferred language with more than 95% accuracy. To operate with such large volumes of data, we needed to develop a java framework

with efficient implementation of the proposed methods. In the thesis we dedicate a special section to the software description and make software publicly available.

Let us mention some open problems and future research directions for the first part. It is worth trying to tune the main parameters of the methods in some automatic fashion. In particular, it will be useful to find a method for joint optimization of the values of α and σ. May be some version of the cross-validation approach is a way to go. In our experiments, we have noticed that choosing the labelled data according to the standard PageRank is better than choosing the labelled data according to the degree. A theoretical justification of this fact is missing. More generally, it would be useful to develop a supervised learning approach where the labelled data is chosen automatically and then the developed semi-supervised learning algorithms are applied. Some initial steps have been done in [10]. However, this research is just in the very beginning and methods with solid theoretical background have to be designed. In [84] it was noticed that incorporating class prior information improves the classification quality. Initial quick experiments with our methods on our datasets, using the suggestion from [84], also show good results. However, a more solid theoretical justification is missing. A research direction, very useful for practice, is to extend our approach to inductive semi-supervised learning [6, 44], which will help us to work with newly arriving out-of-sample data. It would be nice to analyse in full mathematical rigor the classification quality of the proposed semi-supervised learning framework on the clustered preferential attachment model. In the experiments with P2P data we have constructed the user graph and content graph from the original bipartite graph. A natural question is: Could one work directly with the bipartite graph?

The second part of the thesis is dedicated to quick detection of central nodes. There is a clear connection with the first part. Methods for quick detection of central nodes can be used to select quickly high quality labelled points or candidates for labelled points. However, the methods of the second part also have other applications in information retrieval and in network analysis. In the first chapter of the second part we propose and analyse a random walk based method for quick detection of large degree nodes. We show theoretically and by numerical experiments that for large networks the random walk method finds good quality top lists of nodes with high probability and with computational savings of orders of magnitude. In particular, we prove that on the configuration network model, our algorithm has sublinear complexity. We also propose stopping criteria for the random walk method which require very little knowledge about the structure of the network. Then, in the next section we propose and analyse Monte Carlo methods for quick detection of nodes with large values of the Personalized PageRank. As a by-product, we obtain a new matrix form of a central limit theorem for Markov chains. We also show that under fairly general conditions our Monte Carlo approach has sublinear complexity. We illustrate the performance of our approach on large datasets such as English-language Wikipedia. Finally, in the last chapter of the second part we propose a new

centrality measure — Alpha current flow betweenness centrality and its modification, truncated version. The alpha current flow betweenness centrality is a generalization of the current flow betweenness and PageRank. It combines good properties of the current flow betweenness, like taking into account the contribution of short but not only shortest paths, and the computational efficacy of PageRank. We demonstrate that the new centrality measure detects particularly well the network vulnerability.

As the first part, the second part also opens a number of very interesting research directions. Could the proposed method for quick detection of large degree nodes be extended to directed networks? Specifically, what if one wants to detect quickly nodes with the largest incoming or outgoing degrees? Or what if one would like to make a nonparametric or parametric estimation of the tail of the degree distributions in directed/undirected network? How to estimate the maximal size of the top-k list of nodes with the largest degrees or largest values of PageRank, which can be efficiently detected with the random walk based algorithms? In [3, 48] the authors have proposed diffusion type methods for online PageRank computation. It would be interesting to make a theoretical comparative analysis of those diffusion type algorithms and our proposed Monte Carlo algorithms. We have constructed a log-linear complexity algorithm for the alpha current flow betweenness centrality. It would be great also to construct sublinear complexity algorithms for quick detection of nodes with largest values of the alpha current flow betweenness.

10

Résumé en Français

Les méthodes d'apprentissage semi-supervisé constituent une catégorie de méthodes de l'apprentissage automatique qui utilisent des points étiquetés avec les données non étiquetées pour régler le classificateur. Dans la première partie de la thèse, nous proposons une approche d'optimisation généralisée pour les méthodes d'apprentissage semi-supervisé qui donne comme des cas particuliers les méthodes de Laplacien Standard, Laplacien Normalisé et PageRank. En utilisant la théorie de la marche aléatoire, nous fournissons un aperÃ§u sur les différences entre les méthodes d'apprentissage semi-supervisé et donnons des conseils pour le choix des paramètres du noyau et les points étiquetés. Nous avons illustré les résultats théoriques avec des données synthétiques et réelles. Comme un exemple de données réelles, nous considérons la classification des contenus et des utilisateurs dans les systèmes P2P. Cette application montre que la famille des méthodes proposé passe a l'échelle très bien avec le volume de données. La deuxième partie de la thèse est consacrée à la détection rapide des noeuds centraux du réseau. Les algorithmes développés dans la deuxième partie de la thèse peuvent être appliquées pour la sélection des données étiquetées, mais également aux autres applications dans la recherche d'information. Plus précisément, nous proposons des algorithmes randomisé pour la détection rapide des noeuds de grands degrés et des noeuds avec de grandes valeurs de PageRank personnalisé. A la fin de la thèse, nous proposons une nouvelle mesure de centralité, qui généralise à la fois la centralité d'intermédiarité et PageRank. Cette nouvelle mesure est particulièrement bien adapté pour la détection de la vulnérabilité de réseau.

BIBLIOGRAPHY

[1] Hadoop mapreduce software framework, http://hadoop.apache.org/mapreduce/. 2011. 41

[2] Wikipedia article bittorrent (protocol). 2011. 39

[3] S. Abiteboul, M. Preda, and G. Cobena. Adaptive on-line page importance computation. In *Proceedings of the 12th international conference on World Wide Web*, pages 280–290. ACM, 2003. 117

[4] S. Abney. *Semisupervised Learning for Computational Linguistics*. Chapman & Hall/CRC Computer Science & Data Analysis. Taylor & Francis, 2008. 4

[5] D. Aldous and J. Fill. Reversible Markov chains and random walks on graphs. 1999. 28, 94, 97, 99

[6] Y. Altun, D. McAllester, and M. Belkin. Maximum margin semi-supervised learning for structured variables. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 33–40. MIT Press, Cambridge, MA, 2005. 116

[7] R. Andersen, F. Chung, and K. Lang. Using pagerank to locally partition a graph. *Internet Mathematics*, 4(1):35–64, 2007. 12, 31

[8] R. Andersen, F. R. K. Chung, and K. J. Lang. Local graph partitioning using pagerank vectors. In *FOCS*, pages 475–486, 2006. 77

[9] K. Avrachenkov, V. S. Borkar, and D. Nemirovsky. Quasi-stationary distributions as centrality measures for the giant strongly connected component of a reducible graph. pages 3075–3090, 2010. 61

[10] K. Avrachenkov, V. Dobrynin, D. Nemirovsky, S. K. Pham, and E. Smirnova. Pagerank based clustering of hypertext document collections. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 873–874. ACM, 2008. 6, 11, 12, 14, 15, 16, 77, 116

[11] K. Avrachenkov, J. Filar, and P. Howlett. *Analytic Perturbation Theory and its Applications*. SIAM, Philadelphia, 2013. 22, 33

[12] K. Avrachenkov, P. Gonçalves, A. Mishenin, and M. Sokol. Generalized optimization framework for graph-based semi-supervised learning. In *Proceedings of SIAM Conference on Data Mining (SDM'2012), 9 pages*, 2012. 109

[13] K. Avrachenkov and N. Litvak. The effect of new links on Google PageRank. *Stochastic Models*, 22(2):319–332, 2006. 16, 29

[14] K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova. Monte Carlo methods in pagerank computation: When one iteration is sufficient. pages 890–904, 2007. 8, 78, 79, 80

[15] K. Avrachenkov, N. Litvak, D. Nemirovsky, E. Smirnova, and M. Sokol. Monte Carlo methods for top-k personalized pagerank lists and name disambiguation. 2010. 80, 87

[16] K. Avrachenkov, N. Litvak, D. Nemirovsky, E. Smirnova, and M. Sokol. Quick detection of top-k personalized pagerank lists. In *WAW*, pages 50–61, 2011. 72

[17] K. Avrachenkov, B. F. Ribeiro, and D. F. Towsley. Improving random walk estimation accuracy with uniform restarts. In *WAW*, pages 98–109, 2010. 59, 62, 69

[18] B. Bahmani, A. Chowdhury, and A. Goel. Fast incremental and personalized pagerank. pages 173–184, 2010. 78

[19] B. Ball, B. Karrer, and M. E. J. Newman. Efficient and principled method for detecting communities in networks. *Phys. Rev. E*, 84:036103, Sep 2011. 44, 47

[20] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999. 66

[21] C. Barakat, G. Iannaccone, and C. Diot. Ranking flows from sampled traffic. In *CoNEXT*, pages 188–199, 2005. 86

[22] D. Blackwell. Discrete dynamic programming. *Ann. Math. Statist.*, 33:719–726, 1962. 17

[23] P. Boldi, M. Rosa, M. Santini, and S. Vigna. Layered label propagation: a multiresolution coordinate-free ordering for compressing social networks. In *WWW*, pages 587–596, 2011. 59

[24] P. Boldi and S. Vigna. Axioms for centrality. *arXiv:1308.2140*. 104

[25] P. Boldi and S. Vigna. In-core computation of geometric centralities with hyperball: A hundred billion nodes and beyond. *arXiv:1308.2144*. 104

[26] P. Boldi and S. Vigna. The webgraph framework i: compression techniques. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 595–602, New York, NY, USA, 2004. ACM. 42, 59, 80, 109, 110

[27] B. Bollobás. *Modern graph theory*. Springer-Verlag, New York, 1998. 65

[28] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(1994):163–177, 2001. 8, 93, 98

[29] U. Brandes and D. Fleischer. Centrality measures based on current flow. In *Proceedings of the 22nd annual conference on Theoretical Aspects of Computer Science*, pages 533–544, 2005. 93, 95, 97, 98, 102

[30] L.A. Breyer. Markovian page ranking distributions: Some theory and simulations. 2002. 78

[31] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30:107–117, 1998. 94

[32] S. Chakrabarti. Dynamic personalized pagerank in entity-relation graphs. In *WWW*, pages 571–580, 2007. 77

[33] O. Chapelle, B. Schölkopf, and A Zien. *Semi-supervised learning*, volume 2. MIT press Cambridge, 2006. 4, 5, 11, 13

[34] F. Chung and A. Tsiatas. Finding and visualizing graph clusters using pagerank optimization. In Ravi Kumar and Dandapani Sivakumar, editors, *Algorithms and Models for the Web-Graph*, volume 6516 of *LNCS*, pages 86–97. Springer Berlin / Heidelberg, 2010. 12

[35] A. Condon and R. M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Struct. Algorithms*, 18(2):116–140, 2001. 18

[36] C.J. Corrado. The exact joint distribution for the multinomial maximum and minimum and the exact distribution for the multinomial range. Ssrn research report, 2007. 88

[37] S. N. Dorogovtsev, J. F. F. Mendes, and A. Samukhin. Structure of Growing Networks: Exact Solution of the Barabasi–Albert's Model. *Arxiv preprint cond-mat*, January 2000. 59

[38] P.G. Doyle and J.L. Snell. *Random walks and electric networks*. Mathematical Association of America, 1984. 94

[39] D. Fogaras, B. Rácz, K. Csalogány, and T. Sarlós. Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments. pages 333–358, 2005. 78, 80

[40] F. Fouss, K. Francoisse, L. Yen, A. Pirotte, and M. Saerens. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. *Neural Networks*, 31:53–72, 2012. 12, 40

[41] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 1977. 93

[42] A. Gnedin, B. Hansen, and J. Pitman. Notes on the occupancy problem with infinitely many boxes: general asymptotics and power laws. *Probab. Surv.*, 4:146–171, 2007. 88

[43] V. Grolmusz. A note on the pagerank of undirected graphs. *arXiv preprint arXiv:1205.1960*, 2012. 97

[44] Z. Guo, Z. M. Zhang, E. P. Xing, and C. Faloutsos. Semi-supervised learning based on semiparametric regularization. In *SDM*, pages 132–142, 2008. 116

[45] T. H. Haveliwala. Topic-sensitive pagerank. In *Proceedings of the 11th International Conference on World Wide Web (WWW'02)*, pages 517–526, 2002. 6, 29, 77

[46] R. V. D. Hofstad. Random graphs and complex networks. In *In preparation*, 2008. 66

[47] P. Holme, B.J. Kim, C.N. Yoon, and S.K. Han. Attack vulnerability of complex networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 65(5 Pt 2):056109, May 2002. 105

[48] D. Hong and P. Jacquet. Optimizing the eigenvector computation algorithm with diffusion approach. *arXiv preprint arXiv:1206.3177*, 2012. 117

[49] J. J. Hull. A database for handwritten text recognition research. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(5):550–554, 1994. 35

[50] G. Jeh and J. Widom. Scaling personalized web search. In *WWW*, pages 271–279, 2003. 79

[51] N. L. Johnson, S. Kotz, and N. Balakrishnan. *Discrete multivariate distributions*. A Wiley-Interscience publication. Wiley, New York, NY [u.a.], 1997. 83

[52] J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. Undergraduate Texts in Mathematics. Springer, 1976. 18, 83

[53] D. E. Knuth. *The Stanford GraphBase: a platform for combinatorial computing*. ACM, New York, NY, USA, 1993. 19

[54] A. N. Langville and C. D. Meyer. *Google page rank and beyond*. Princeton University Press, 2006. 16

[55] A.N. Langville and C.D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, Princeton, NJ, 2006. 79

[56] S. Le Blond, A. Legout, F. Lefessant, W. Dabbous, and M. A. Kaafar. Spying the world from your laptop: identifying and profiling content providers and big downloaders in bittorrent. In *Proceedings of the 3rd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more*, LEET'10, pages 4–4, Berkeley, CA, USA, 2010. USENIX Association. 39, 41

[57] W. Li, M. Canini, A. W. Moore, and R. Bolla. Efficient application identification and the temporal and spatial stability of classification schema. *Comput. Netw.*, 53:790–809, April 2009. 39

[58] W. Li and A. W. Moore. A machine learning approach for efficient traffic classification. In *Proceedings of the 2007 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pages 310–317, Washington, DC, USA, 2007. IEEE Computer Society. 39

[59] D. Liben-Nowell and J. M. Kleinberg. The link prediction problem for social networks. In *CIKM*, pages 556–559, 2003. 77

[60] L. Lovász. Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46, 1993. 28

[61] D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, and S.M. Dawson. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405, September 2003. 101

[62] A. S. Maiya and T. Y. Berger-Wolf. Online sampling of high centrality individuals in social networks. In *PAKDD (1)*, pages 91–98, 2010. 57

[63] A. Mantrach, N. Van Zeebroeck, P. Francq, M. Shimbo, H. Bersini, and M. Saerens. Semi-supervised classification and betweenness computation on large, sparse, directed graphs. *Pattern recognition*, 44(6):1212–1224, 2011. 8, 94

[64] G. Matthys and J. Beirlant. Estimating the extreme value index and high quantiles with exponential regression models. *Statistica Sinica*, 13(3):853–880, 2003. 66

[65] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, Cambridge, 2005. 58, 70

[66] C. B. Moler. *Numerical Computing with MATLAB*. 2004. 16

[67] A. A. Moreira, J. S. Andrade, and L. A. Nunes. Extremum statistics in scale-free network models. *Phys. Rev. Lett.*, 89:268703, Dec 2002. 67

[68] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1995. 58

[69] D. Nemirovsky. Tensor approach to mixed high-order moments of absorbing markov chains. *Linear Algebra and its Applications*, 2011. 84

[70] M. E. J. Newman. A measure of betweenness centrality based on random walks. *Social networks*, pages 1–15, 2005. 8, 93, 97, 98, 102

[71] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, 2004. 19

[72] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120. 77

[73] M. Pietrzyk, J. Costeux, G. Urvoy-Keller, and T. En-Najjary. Challenging statistical classification for operational usage: the adsl case. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, IMC '09, pages 122–135, New York, NY, USA, 2009. ACM. 39

[74] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994. 17

[75] E. Smirnova, K. Avrachenkov, and B. Trousse. Using web graph structure for person name disambiguation. In *CLEF (Notebook Papers/LABs/Workshops)*, 2010. 77, 80

[76] K. Tanabe and M. Sagae. An exact Cholesky decomposition and the generalized inverse of the variance-covariance matrix of the multinomial distribution with applications. *Journal of the Royal Statistical Society. Series B (Methodological)*, 54(1):211–219, 1992. 84

[77] K. Voevodski, S. H. Teng, and Y. Xia. Spectral affinity in protein networks. *BMC Systems Biology*, 3(1):112+, 2009. 77

[78] Y. Wu, P. A. Ruibal, M. K. Jones, and C. Genzmer. Using social signals to identify unauthorized content on a social networking system, June 13 2013. US Patent 20,130,152,211. 3

[79] L. Yeon-Sup, D. S. Menasche, B. Ribeiro, D. Towsley, and P. Basu. Online estimating the k central nodes of a network. In *Proceedings of the 2011 IEEE Network Science Workshop*, NSW '11, pages 118–122, Washington, DC, USA, 2011. IEEE Computer Society. 57

[80] G. Yin and Q. Zhang. *Continuous-time Markov chains and applications: a singular perturbation approach*. Applications of mathematics. Springer, 1998. 22

[81] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*, pages 321–328. MIT Press, 2004. 5, 11, 13, 15

[82] D. Zhou and C. J. C. Burges. Spectral clustering and transductive learning with multiple views. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 1159–1166. ACM, 2007. 5, 11, 13, 15

[83] X. Zhu. Semi-supervised learning literature survey, 2006. 4

[84] X. Zhu, Z. Ghahramani, J. Lafferty, et al. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, volume 3, pages 912–919, 2003. 116

[85] X. Zhu and A. B. Goldberg. Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 3(1):1–130, 2009. 4, 11

[86] V. Zlatic, M. Bozicevic, H. Stefancic, and M. Domazet. Wikipedias: Collaborative web-based encyclopedias as complex networks. *Physical Review E*, 74:016115, 2006. 80