Empirical Comparison of Graph-based Recommendation Engines for an Apps Ecosystem

Luis F. Chiroque, Héctor Cordobés, Antonio Fernández Anta, Rafael A. García Leiva, Philippe Morere, Lorenzo Ornella, Fernando Pérez, Agustín Santos

***Resumen* —Recommendation engines (RE) are becoming highly popular, e.g., in the area of e-commerce. A RE offers new items (products or content) to users based on their profile and historical data. The most popular algorithms used in RE are based on collaborative filtering. This technique makes recommendations based on the past behavior of other users and the similarity between users and items.**

In this paper we have evaluated the performance of several RE based on the properties of the networks formed by users and items. The RE use in a novel way graph theoretic concepts like edges weights or network flow. The evaluation has been conducted in a real environment (ecosystem) for recommending apps to smartphone users. The analysis of the results allows concluding that the effectiveness of a RE can be improved if the age of the data, and if a global view of the data is considered. It also shows that graph-based RE are effective, but more experiments are required for a more accurate characterization of their properties.

***Palabras clave*— Recommendation engines, smartphone apps, graph theory, collaborative filtering, flow algorithms.**

I. Introduction

A. Motivation

It is becoming very common in online platforms (shopping websites, online newspapers, online social networks, smartphone apps, etc.) to recommend items to the users that will (hopefully) be of their interest. This trend is becoming so general that Anderson predicted that we are “leaving the age of information and entering the age of recommendation” [4]. The items to recommend are selected by a recommendation engine (RE) that typically leverages the user profile, the context, and historical data. The RE typically has a catalog of items from which to choose its recommendation, and there are spaces in the online platform viewing area in which the recommended product is presented. The context of the user typically includes its past navigation history, including the current viewing context, which may involve a product (e.g., in a shopping website), a piece of news (e.g., in an online newspaper), a user profile (e.g., in an online social network), or the application that is being executed (e.g., in a smartphone).

Recently, the most popular algorithms used in RE are based on collaborative filtering [15]. This technique makes recommendations based on the historical data of all the users and the estimated similarity between them. Typical metrics used for the computation of customers' similarity include Pearson correlation coefficient, adjusted cosine similarity, Spearman's rank correlation coefficient, and mean squared difference.

In parallel with the advances in RE algorithms, we have observed that graph theory and network analysis has been useful in different contexts to extract information from data. This information is not an explicit part of the data, but it is implicitly contained in the underline structure. Examples of this approach are the use of pagerank to identify the most relevant web pages [8], or a recent use we have made of graphs to classify tweets [9]. We believe that graph theory and network concepts can also be useful in the context of recommendation.

B. Contributions

In this paper we present an exploratory work on using graphs to build RE. We have devised, developed, and evaluated RE based on collaborative filtering to promote an ecosystem of smartphone apps. In this ecosystem, the users of the apps get banners advertising other apps that they have not installed (yet). The objective of the RE is maximizing the click-through rate (CTR) of users in these banners, and maximizing the installation of new apps. In addition we have devised one particular RE to promote a specific subset of apps. The proposed RE create models of the ecosystem as networks formed by apps, and use graph theoretic concepts like edges weights or network flow.

The performance of the RE proposed has been evaluated in a real apps ecosystem. Several years’ worth of historical data has been used to create the networks that model the

Manuscript received January 31, 2015. This work was supported in part by the SOCAM research project, Spanish Ministry of Industry, Energy and Tourism.

L. F. Chiroque, H. Cordobés, A. Fernández Anta, R. A. García Leiva, P. Morere, and A. Santos are with IMDEA Networks Institute, Madrid, Spain (e-mail: luisfelipe.nunez, hector.cordobes, antonio.fernandez, rafael.garcia, philippe.morere, agustin.santos@imdea.org).

L. Ornella is with Politecnico di Torino, Turin, Italy (e-mail: lorenzo.ornella@studenti.polito.it).

F. Pérez is with U-tad, Madrid, Spain (corresponding author, e-mail: fernando.perez@u-tad.com).

ecosystem. Then, using them, the different RE were put to work with real users for about a week. The analysis of the results obtained has shown big (statistically significant) differences in CTR and installation success of the different RE. The results allow concluding that the effectiveness of a RE can be improved if the age of the data, and if a global view of the data is considered. It also shows that graph-based RE are effective. However, some of the results are puzzling, and hence more experiments are required for a more accurate characterization of the properties of the proposed RE.

C. Structure

The rest of the paper is structured as follows. In Section II we present the problem to be solved. In Section III we describe the RE we have proposed and that will be evaluated in this paper, with the underlying graph they use. In Section IV we present the experiment we have conducted, the results obtained, and some discussion on them. Section V presents previous work related to this paper. Finally, Section VI concludes the paper.

II. Problem Statement

As described, we have a *smartphone app ecosystem.* In this ecosystem, a user that is running an app (called the *publisher*), gets *banners* advertising other apps of the ecosystem it has not installed yet. The objective is to devise a RE that tells the system which app to advertise to a given user at a given time, possibly as a function of the user and the publisher, in order to achieve one (or more) of the following objectives.

- *CTR Maximization:* The objective is to maximize the number of times the user clicks in the banner to get more information about the apps advertised.
- *Installations Maximization:* The objective is to maximize the number of times the user installs the app advertised.
- *Targeted Promotion:* The objective is to maximize the number of times users install a preselected set of apps to be promoted.

An initial hypothesis we will make is that, once a user has clicked in a banner, the probability of installing the corresponding app is roughly same. This has made us concentrate initially in RE for the CTR Maximization and Targeted Promotion objectives. (As will be seen from the results obtained, this initial hypothesis needs some revision.)

III. Recommendation Engines Proposed

In this section we describe the recommendation engines we have proposed and evaluated in this paper. In order to describe them, we build graphs from historical user data that convey the essential information that is required by the corresponding RE. Hence, we start describing the graphs we need and use, and then we give the algorithms used by the RE to select an app to advertise.

A. Apps Graphs

All the graphs used in this work will have the set of apps A of the ecosystem as vertices. Moreover, all of them are weighted graphs, and the main difference among them is the weights that are allocated to edges. The graphs used are the following.

Shared Users (SU) Graph. The SU graph is an undirected weighted graph $G_{SU}=(A,E,w)$, where $E=\{\{i,j\}: i,j \in A\}$ and the weight $w(e)$ of an edge $e=\{i,j\} \in E$ is the number of users of the ecosystem that have currently both apps i and j installed.

Aged Shared Users (ASU) Graph. The ASU graph is an undirected weighted graph $G_{ASU}=(A,E,w)$, where $E=\{\{i,j\}: i,j \in A\}$ like in G_{SU}. The difference in this case is that the contribution to the weight $w(e)$ of an edge $e=\{i,j\} \in E$ of a user (that has currently both apps i and i installed) is a function of the time the user has had the apps installed. In particular, let U be the set of all users and $U(a) \subseteq U$ be the set of users that have app $a \in A$ installed. Also, let $age(u,a)$ be the time since user $u \in U(a)$ installed app $a \in A$ (in some suitable units). Then,

$$w(\{i,j\}) = \sum_{u \in U(i) \cap U(j)} \delta^{\min\{age(u,i), age(u,j)\}},$$

where $\delta \leq 1$ is the decay factor. (The intuition is that users that installed an app long time ago are less important for the app.)

CTR Graph. The CTR graph is a *directed* weighted graph $G_{CTR}=(A,E,w)$, where $E = A \times A$ and the weight $w(e)$ of an edge $e=(i,j) \in E$ is the CTR observed when banners with app j are presented to the users with app i as publisher.

B. Recommendation Algorithms

Using the above graphs we can describe now the RE considered in this work.

Shared Users.

Let us consider the SU graph described above. Assuming the publisher app is i, the app recommended j is the one that has the edge with i of largest weight. I.e., $j=argmax_k(w(\{i,k\}):k \in A)$. In this case, this means that j is the app with the largest number of common users with i.

The approach of this algorithm is not new, and it is among the first ideas one may think of when resigning recommendation algorithms.

Collaborative Filtering.

This algorithm also uses the SU graph. Given the user to which the banner will be presented, and the set I of applications the user has already installed, the app j recommended is the one that has a largest aggregate weight with those in set I. I.e., $j=argmax_k(\Sigma_{i \in I}\, w(\{i,k\}):k \in A)$.

Again, this approach is not very novel, since it is common to many collaborative filtering algorithms to use some linear algebra approach that can achieve similar results as this one. For instance, considering the weights of the SU graph as a matrix W, and the applications already installed by the used as a vector v, the algorithm proposed would recommend the app that corresponds to the largest element of the vector $v^T W$.

Aged Shared Users.

This algorithm applies the same process as Shared Users, but in the ASU graph. As far as we know this algorithm is new.

Aged Collaborative Filtering.

This algorithm applies the same process as Collaborative Filtering, but in the ASU graph. As far as we know this algorithm is also new.

Maxflow.

This algorithm uses the CTR graph with the objective of promoting a preselected subset *P* of apps. The algorithm takes the publisher app *i* and solves a flow maximization problem [11] from *i* to each of the apps in *P*, where the weight of each link is considered its capacity. Then, it recommends the neighbor of *i* whose aggregated flow is the largest. I.e., imagine that the solution of the maximum flow problem from *i* to $a \in P$ sends $f(a,k)$ units of flow across link (i,k). Then, the recommended app is $j=argmax_k(\Sigma_{a \in P} f(a,k):k \in A)$.

To our knowledge, the Maxflow RE is also new. The intuition behind it is that instead of directly promoting the apps in *P* it is better to promote those that will drive the user to them.

In addition to the 5 RE described, we will consider for reference two trivial algorithms.

Random.

This algorithm recommends an app at random using a uniform distribution over the set of available applications. As we just said, the goal of the random RE is to have a reference with which all the other RE can be compared.

Static Promotion.

This algorithm always recommends one of the applications of the set *P* to be promoted (chosen uniformly at random). It does not depend on the user installed applications, nor the publisher.

IV. Performance Evaluation

In this section we describe the experiment we have conducted in order to evaluate and compare the RE proposed. Then, the results obtained in the experiment are presented and briefly analyzed.

A. Implementation of the Experiment

As mentioned previously, the evaluation of the RE has been done in a real apps ecosystem. This ecosystem is formed by roughly 300 apps with more than 4 million users.

To build the graphs used by the RE and described in the previous section, we have used more than 3 years worth of data. This adds to more than 100 GiB of historical data structure in more than 1.4 billion records. This data has been processed with Big Data technologies (Hadoop, Pig, Hbase [5-7]) in the Amazon Elastic Map Reduce [2] environment. The processing involved cleaning the historic data generated a clean dataset of more than 700 million records of events, containing the user, the publisher, the app advertised (in the banner), the action associated to the event (add, click), and the timestamp.

From the clean dataset just obtained, the above-described graphs were built. The construction of the aged graph ASU used a value of $\delta=0.95$ and the age is measured in units of weeks. It is important to note that the historical data to which we had access did not record explicitly the installation of the apps. The fact that a user had an app installed was extracted from the data because the app appeared as publisher in some event.

Once the graphs were ready, we run an experiment in the real system for about a week (from Jun 2nd, 2014 to Jun 10th, 2014). In this experiment, the different RE recommended the apps shown in banners to the users. In order to avoid cross interference, the same RE generated all the banners for the same user. For the targeted promotions RE proposed (Maxflow and Static Promotion) a manually selected set of 5 apps were chosen to be promoted. At the end of the experiment, banner were shown to more than 300,000 users, and each RE had done more than 130,000 recommendations.

After the experiment, the data of number of banners recommended by each RE, the number of clicks by the user, and the number of apps installed was obtained. It is important to note that the data obtained was cleaned. For instance, multiple clicks associated to the same banner where counted only once. Regarding installations, we assumed that a banner had caused the installation of an app if the app was used (by the user) within 72 hours after the banner was shown.

B. Experiment Results

The basic results obtained in the experiment are presented in Table I. For each RE the table shows the total number of banners that used the RE for recommendation, the number of banners on which the user clicked, and the click through rate, CTR, which is the ratio of the former two values. Additionally, the number of installations from the banner is also shown. Finally, we present two metrics, installation to banners rate (IBR) and installation to clicks rate (IBR), which are the ratio of the number of installations versus the number of banners and the number of clicks, respectively.

Table I: Results of the experiment

RE	Banners	Clicks	CTR (%)	Installs	IBR (%)	ICR (%)
Random	140894	1993	1.41	126	0.09	6.32
Shared Users	133818	2095	1.57	299	0.22	14.27
Aged Shared Users	139417	2258	1.62	390	0.28	17.27
Collaborative Filtering	134790	1966	1.46	329	0.24	16.73
Aged Collaborative Filtering	133623	2204	1.65	375	0.28	17.01
Static Promotion	138922	1929	1.39	215	0.15	11.15
Maxflow	140858	2302	1.63	290	0.21	12.60

As can be seen from the results presented, the CTR observed is different for different RE. Table II presents a comparison of the differences of the CTR achieved by the RE. In each entry of the table it is show (in percentage) the increase in CTR achieved if using the RE of the column instead of the RE of the row. When this number is negative the CTR in fact decreases, and the value is marked in red.

Table II: Increase of the CTR (in percentage) when using one RE (column) versus another (row)

	Random	Shared Users	Aged Shared Users	Collaborative Filtering	Aged Collaborative Filtering	Static Promotion	MaxFlow
Random		10.68	14.50	3.11	16.60	-1.84	15.53
Shared Users	-9.65		3.45	-6.83	5.36	-11.31	4.39
Aged Shared Users	-12.66	-3.34		-9.94	1.84	-14.27	0.91
Collaborative Filtering	-3.02	7.34	11.04		13.08	-4.80	12.05
Aged Collaborative Filtering	-14.24	-5.08	-1.81	-11.57		-15.82	-0.92
Static Promotion	1.87	12.75	16.64	5.04	18.79		17.70
MaxFlow	-13.45	-4.20	-0.90	-10.75	0.93	-15.04	

Given these differences CTR between the RE used, we want to determine if they are statistically significant. For that, we have computed a *z*-test [19] to compare the CTR of each pair of RE. For a given pair of RE the null hypothesis is that both populations are extracted from equal distributions (and hence the differences are simply due to statistical noise). The alternative hypothesis is then that the distributions are different. We compute the *z* value for two RE (numbered 1 and 2) as

$$z = \frac{p_1 - p_2}{\sqrt{p(1-p)}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}},$$

where n_i is the number of banners of RE *i*, $p_i = x_i / n_i$ is the ratio between the number of clicks x_i and the number of banners, and *p* is the pool population, defined as

$$p = \frac{x_1 + x_2}{n_1 + n_2}.$$

The results of all the z-tests are shown in Table III. Using an alpha value of 0.05, we accept the null hypothesis if the z value is within the interval [-1.96,1.96], and reject it otherwise, with a confidence of 0.95. If the null hypothesis is rejected for a pair of RE, it means that with a probability of at least 0.95 the CTR of one is larger than the CTR of the other. In Table III the pairs of RE for which the null hypothesis is rejected have white background, and we use yellow background for the cases in which the null hypothesis cannot be rejected.

Finally, we present the results of the different RE as promoters of specific apps. As described above, we have devised a RE, Maxflow, specifically for targeted promotion of apps, and used it for promoting the 5 chosen apps.

Table III: *z*-values computed for the CTR of each pair of RE.

	Random	Shared Users	Aged Shared Users	Collaborative Filtering	Aged Collaborative Filtering	Static Promotion
Shared Users	3.267683					
Aged Shared Users	4.441854	1.127828				
Collaborative Filtering	0.9712663	-2.272151	-3.423103			
Aged Collaborative Filtering	5.013088	1.724149	0.6142521	3.997578		
Static Promotion	-0.5847134	-3.833009	-5.007191	-1.546221	-5.570983	
MaxFlow	4.759739	1.434164	0.3069088	3.734905	-0.3121384	5.324833

In addition, as mentioned, we implemented a trivial RE, Static Promotion, which only recommends the 5 apps to be promoted. In Table IV we present the number of installations that these RE achieved for each of the four apps to be promoted, numbered from 1 to 5. For comparison, we also show the numbers of installations achieved with the other RE.

Table IV: Installations of the 5 apps promoted

RE	App 1	App 2	App 3	App 4	App 5
Random	2	0	4	0	4
Shared Users	18	1	13	45	31
Aged Shared Users	13	0	15	50	42
Collaborative Filtering	7	0	22	36	39
Aged Collaborative Filtering	13	0	22	47	43
Static Promotion	35	17	38	70	47
Maxflow	0	0	2	11	46

C. *Discussion*

Table I shows significant differences between the RE used. The first fact to note is that, as expected, both Random and Static Promotion have very low CTR and IBR. All the other algorithms have a CTR that is at least 3% higher than Random and 5% higher than Static Promotion (see Table II). The difference in IBR is even higher, were every RE achieves at least a 133% increase over Random and 40% over Static Promotion.

Comparing the CTR of the rest of RE, there is a significant difference between Shared Users versus Aged Shared Users, and Collaborative Filtering versus Aged Collaborative Filtering. This difference leads to conjecture that the preferences of the users change over time. This is the reason why the RE that take that into account this evolution behave well. Somewhat surprising is the high CTR achieved by Maxflow, which has the second largest CTR, since the objective of this RE is not to maximize the CTR.

Table III shows that the differences between the CTR are statistically significant many cases. In particular, in terms of CTR, the *z*-test divides the RE into two groups. One group has CTR that is statistically smaller than the other. Random, Static Promotion, and Collaborative Filtering form the group of low CTR. The group of high CTR includes Shared Users, Aged Shared Users, Aged Collaborative Filtering, and Maxflow. Observe that a larger alpha value in the *z*-test would differentiate the RE further.

Looking at the ICR columns in Table I, we can see that the values in the column differ significantly. This disproves our initial hypothesis that, once a user clicks in a banner, she has a similar probability of installing the app. The conclusion is that it is not enough to aim at maximizing CTR if the objective is to get app installations. For instance this causes that Maxflow is the RE with the lowest IBR from those not for reference.

Finally, regarding targeted promotion, in Table IV we can observe that Maxflow achieves a low number of installations for the 5 promoted apps, especially compared with Static Promotion (but even versus all the other RE except Random). This result is disappointing, and it requires further study. Our conjecture is that the experiment conducted was too short to observe the effect of Maxflow, which promotes the apps that lead to other apps. Other lines to explore are the modification of Maxflow in two ways. First, Maxflow must be tested using a IBR graph instead of the CTR graph (since, from a previous discussion CTR is not the critical metric if we want installations). Second, the graph used by Maxflow must consider aging, since as we have observed this is an important aspect of the data. In any case, another conclusion we obtain from Table IV is that using Static Promotion for targeted promotion of apps seems like a valid option.

V. Related Work

The most common approaches to the recommendation problem can be grouped into three types.

- Collaborative filtering [15]: In this approach users are represented by an *N*-dimensional vector of items, and the recommender looks for users who have similar rating patterns as the target user. Then, it uses the ratings from those like-minded users to make a recommendation for the target user.
- Cluster models: This approach divides the customer base into many segments, and treats the recommendation task as a classification problem. Segments are created using a clustering, or some other unsupervised learning algorithm.
- Search-based methods: In this approach, given the target user's purchased and rated items, the algorithm constructs a search query to find other popular items by the same author, artist, or director, or with similar keywords or topics.

As an example, Amazon uses its own recommendation algorithm, called item-to-item collaborative filtering [17], to personalize the online store for each customer. The algorithm is focused in finding similar items, not similar customers, and hence it scales independently of the number of customers. However, the challenge is to make it scalable with the number of items in the product catalog.

Most of the collaborative filtering algorithms we have found in the literature assume that user preferences remain stable and consistent over time [14]. We believe this is not generally the case, and our conjecture is supported by the fact that in our experiment the RE that considered aging performed very well.

Methodologies for the evaluation of RE have been proposed in [18] and [12]. Other aspects, like advertising effectiveness and Return of Investment (ROI) on social networks, have been a big topic of discussion for advertisers in the past decade [3]. ROI has been typically measured through econometric models that measure the impact of varying levels of advertising (Gross Ratings Points, GRP) on sales, on purchases decision, and choices made. (Finding improved methods of measuring ROI is still an important area of research.) A classical introductory paper is due to Danaher and Rust [10]. Taylor [20] has summarized the current focus of research on advertising.

This is not the first paper that presents approaches based on graphs for recommendation systems. Huang et al. [13] proposed to build a bipartite graph of users and items, where user vertices are connected with item vertices if the user bought or gave a good evaluation to the item. The authors estimate the interest of a given user in a give item by aggregating the weights of short path between the user and the item in the graph. Lien and Phuong [16] extend the users-items bipartite graph with weights representing the evaluation the users gave to the items. Regarding flows, Adomavicius and Kwon [1] used a maximum flow algorithm for maximizing the diversity of the recommendations (instead of improving the recommendation accuracy as we do).

VI. Conclusions

We have presented a collection of graph-based recommendation engines, and have tested them in a real ecosystem of smartphone apps. The results obtained drive us to conjecture that using graphs for recommendation is a promising line of research. However, more experiments are needed in order to verify or disprove this conjecture.

In this work we have built recommendation engines that used graphs of items. We believe that graphs of users could also be very useful for recommendation. However, these graphs tend to be must larger (of several million nodes in our real system, versus a few hundreds of item), and processing them requires using more powerful computational systems and developing scalable algorithms.

References

[1] G. Adomavicius, Y. Kwon, “Maximizing Aggregate Recommendation Diversity: A Graph-Theoretic Approach,” in *Workshop on Novelty and Diversity in Recommender Systems,* 2011.

[2] Amazon Web Services, Inc., *Amazon Elastic MapReduce,* 2014. [Online http://aws.amazon.com/elasticmapreduce/; accessed 6-August-2014].

[3] T. Ambler, J. H. Robers, “Assessing marketing performance: don't settle for a silver metric,” *Journal of Marketing Management,* 24 (7,8), 733-750, 2008.

[4] C. Anderson, “The Long Tail: Why the Future of Business Is Selling Less of More,” *Hyperion,* 2006.

[5] The Apache Software Foundation, *The Apache Hadoop project,* 2014. [Online http://hadoop.apache.org/; accessed 6-August-2014].

[6] The Apache Software Foundation, *Apache HBase,* 2014. [Online http://hbase.apache.org/; accessed 6-August-2014].

[7] The Apache Software Foundation, *Apache Pig,* 2014. [Online http://pig.apache.org/; accessed 6-August-2014].

[8] S. Brin, L. Page, “The Anatomy of a Large-Scale Hypertextual Web Search Engine,” *Computer Networks* 30(1-7): 107-117, 1998.

[9] H. Cordobés, A. Fernández Anta, L. F. Chiroque, F. Pérez, T. Redondo, A. Santos, “Graph-based Techniques for Topic Classification of Tweets in Spanish,” *IJIMAI* 2(5): 31-37, 2014.

[10] P. J. Danaher, R. T. Rust, “Determining the optimal return on investment for an advertising campaign,” *European Journal of Operational Research,* 95:511-521, 1996.

[11] L. R. Ford Jr., D. R. Fulkerson, *Flows in Networks*, Princeton University Press, 1962.

[12] J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, “Evaluating collaborative filtering recommender systems,” *ACM Transactions on Information Systems,* vol 22(1), pp. 5-53, 2004 .

[13] Z. Huang, H. Chen, D. Zeng, “Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering”, *ACM Transactions on Information Systems,* vol. 22(1), pp. 116–142, 2004.

[14] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, *Recommender Systems: An Introduction,* Cambridge, 2010.

[15] Y. Koren, R. M. Bell, “Advances in collaborative filtering,” in *Recommender Systems Handbook,* Springer, pp. 145–186, 2011.

[16] D. T. Lien, N. D. Phuong, “Collaborative filtering with a graph-based similarity measure,” in *International Conference on Computing, Management and Telecommunications (ComManTel),* 2014.

[17] G. Linden, B. Smith, J. York, “Amazon.com recommendations: Item-to-item collaborative filtering,” *IEEE Internet Computing,* 7(1), 76-80, 2003.

[18] G. Shani, A. Gunawardana, “Evaluating Recommendation Systems,” in *Recommender Systems Handbook,* Springer, pp. 257-297, 2011.

[19] R.C. Sprinthall, Basic Statistical Analysis, 9th Edition, *Pearson Education Group,* 2011.

[20] C. R. Taylor, “Hot topics in advertising research,” *International Journal of Advertising,* 32(1), 2013.

Luis F. Chiroque obtained his B.Sc. degree in Telematics Engineering from the Polytechnic University of Madrid. Currently, he is studying a Master's Degree in Mathematical Engineering at University Carlos III of Madrid and he is a PhD student at IMDEA Networks, where he previously worked in the SOCAM project. His main research interest areas are graph theory, social networks and big data.

Héctor Cordobés obtained his MSc in Telecommunications in 2003 from Universidad Carlos III de Madrid. From that year he worked for Motorola, participating in international projects such as Motorola Soft Switch, Motorola IMS Core Network, High-Availability Computing Platform or Kreatel IPTV Platform, as Senior System Architect and Developer. In 2013 he joined IMDEA Networks as a Research Engineer, working in Natural Language Processing and Data Analysis projects.

Antonio Fernández Anta is a Research Professor at IMDEA Networks. Previously he was a Full Professor at the Universidad Rey Juan Carlos (URJC) and was on the Faculty of the Universidad Politécnica de Madrid (UPM), where we received an award for his research productivity. He was a postdoc at MIT from 1995 to 1997. He has more than 20 years of research experience, with a productivity of more than 5 papers per year on average. He is Chair of the Steering Committee of DISC and has served in the TPC of numerous conferences and workshops. He is a senior member of the IEEE and the ACM. He received his M.Sc. and Ph.D. from the University of Louisiana in 1992 and 1994, respectively. He completed his undergraduate studies at the UPM, having received awards at the university and national level for his academic performance.

Rafael A. García Leiva has a Bachelors degree in Computer Science by the University of Córdoba (Spain), a Master degree in Computational Sciences by the Univesity of Amsterdam, and a Diploma of Advanced Studies in Telematics by the Universidad Autónoma de Madrid. He worked during four years in the University of Córdoba as research assistant and scientific programmer, in the areas of Systems and Networks, Geographical Information Systems and Remote Senging. He worked during three years in the Universidad Autónoma de Madrid as research engineer in the area of High Energy Physics. He worked during three years as R&D manager at Andago Ingeniería, coordinating the R&D activities of the company. In 2008 he founded his own company, Entropy Computational Services, where he worked during five years in the areas of Social Networks, Mobile Applications, and Quantitative Trading. In 2014 he joined to the Institute IMDEA Networks, as research engineer, working in the area of Big Data and Machine Learning.

Philippe Morere is a member of the Research Support team at Institute IMDEA Networks. He works as a Research Engineer under the supervision of Antonio Fernández Anta, Research Associate Professor at the Institute. Prior to his incorporation to IMDEA Networks, Philippe worked as an Internship Student for Ritsumeikan University (Japan), where he was researching on embedded vision systems for intelligent robotic control. He obtained a B.Sc. degree in Engineering Sciences from the engineering school Enseirb-Matmeca, University of Bordeaux 1 (France). A few years later, he went on to pursue a M.Sc. degree in Computer Sciences & Telecommunications also at Enseirb-Matmeca.

Lorenzo Ornella received in 2014 his M.Sc. Degree in Computer and Communication Networks Engineering from the Politecnico di Torino (Turin, Italy). He previously obtained a Laurea in Ingegneria Delle Telecomunicazioni. He collaborated with IMDEA Networks and Zed from November 2013 until May 2014.

Fernando Pérez García, is Big Data & Advanced Analytics Services manager at Zed. He was before at IBM, as a Data Specialist, in the Business Analytics & Optimization service line, developing Business Intelligence and Data Mining projects for main Spanish banks and carriers. Fernando is an entrepreneur, Android and Graphs enthusiast. Creator of Nautka Apps, Fernando has developed applications with over 150K users, providing natural language processing technologies to final users. Fernando is the inventor of the US Patent 7289025: "Method and system for securing an electronic device", on October 2007. Fernando completed his undergraduate studies (Licenciado en Informática) at the Universidad Pontificia de Salamanca in 1997.

Agustín Santos, PhD in Computer Science from Universidad Rey Juan Carlos de Madrid. He currently works at IMDEA Networks in several R+D projects. He previously developed his professional activities in the private sector. His broad experience covers technological base companies for which he has been founding partner. He has also held the positions of manager and chief director. His interests are focused in the fields of Distributed Systems, Big Data, Simulation, and Natural Language Processing.