(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
29 October 2015 (29.10.2015)

WIPO | PCT

(10) International Publication Number
WO 2015/162286 A1

(51) International Patent Classification:
G06Q 50/00 (2012.01)

(21) International Application Number:
PCT/EP2015/059030

(22) International Filing Date:
27 April 2015 (27.04.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
14166040.7 25 April 2014 (25.04.2014) EP

(71) Applicant: **NEC EUROPE LTD.** [DE/DE]; Kurfürsten-Anlage 36, 69115 Heidelberg (DE).

(72) Inventors: **JACOBS, Tobias**; Chamissostraße 1-3, 68167 Mannheim (DE). **CIRILLO, Flavio**; Via Pier Paolo Pasolini 59, 81000 Caserta (IT). **KOVACS, Ernoe**; Kleinhohenheimerstraße 20, 70619 Stuttgart (DE).

(74) Agent: **ULLRICH & NAUMANN**; Schneidmühlstraße 21, 69115 Heidelberg (DE).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

(54) **Title**: METHOD FOR DETECTING INTERACTIONS BETWEEN USERS OF A NETWORK

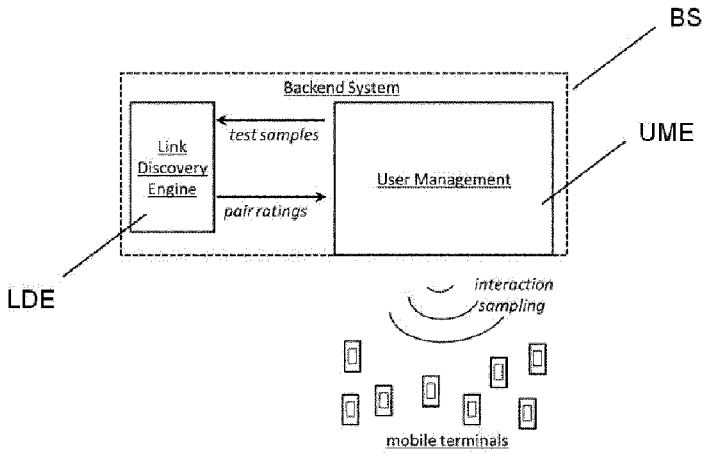


Fig. 1

(57) **Abstract**: A method for detecting interactions between users of a network, performed in a memory available to one or more computation devices comprising the steps of • a) Determining a maximum number of tests for detecting interactions • b) Performing a first number of tests by random sampling users for detecting interactions between them if any • c) Determining interactions between users by evaluating said first number of tests • d) Evaluating said determined interactions • e) Performing one or more further tests based on the prior evaluated interactions • f) Determining interactions between users using evaluations of said performed further tests • g) Perform steps d)-g) again as long as the maximum number of tests is not exceeded.

WO 2015/162286 A1

METHOD FOR DETECTING INTERACTIONS BETWEEN USERS OF A NETWORK

The European patent application EP 14 166 040.7 is herein incorporated by reference.

The present invention relates to a method for detecting interactions between users of a network, performed in a memory available to one or more computation devices.

The present invention further relates to a system for detecting interactions between users of a network, comprising one or more computation devices.

Although applicable to any kind of network in general, the present invention will be described with regard to social networks.

Although applicable in general to any kind of users, the present invention will be described with regard to mobile users.

Although applicable to any kind of system in general, the present invention will be described with regard to a backend system.

Social networks have become more and more interesting for many users in the last years for various purposes, for example for contacting old friends or to establish business contacts or the like. For a service provider it might be interesting to extract information from such a social network whose structure is unknown to him.

One of the main issues a service provider might be interested are interactions between users of the social network, interactions of users with services and/or interactions between services. A problem arises when the structure of such a network is not immediately accessible to applications. For example third-party providers of social services which use information from one or more social networks they are not owning need to retrieve structural network information for

example via external interfaces. Thus testing social links between service users cannot be done arbitrarily frequently.

In other cases testing of social links requires intensive computational large amount of data for example when involve data mining methods are applied. Also here the frequency of testing interactions needs to be used effectively. Cases of expensive discovery of links have also been shown for other kind of networks like molecular interacting networks as disclosed in the non-patent literature of Linyuan Lü and Tao Zhou, "Link prediction in complex networks: A survey", Physica A: Statistical Mechanics and its Applications, 390(6):1150-1170, 2011. Here efforts from various research areas regarding link prediction are summarized and categorized. Furthermore an overview of applications is given and a methodology is described to evaluate the performance of link prediction algorithms.

In the non-patent literature of David Liben-Nowell and Jon Kleinberg, "The link-prediction problem for social networks", Journal of the American society for information science and technology, 58(7):1019-1031, 2007 a broad range of methods for predicting links in co-authorship networks and their evaluation is disclosed. In addition the experimental setup also includes methods for pre-processing an input graph before an actual prediction is done.

In the non-patent literature of Mohammed Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki, "Link prediction using supervised learning", in SDM06: Workshop on Link Analysis, Counter-terrorism and Security, 2006 a set of features partly with scores, used in the non-patent literature of David Liben-Nowell and Jon Kleinberg, "The link-prediction problem for social networks", Journal of the American society for information science and technology, 58(7):1019-1031, 2007, partly features extracted from information other than the network structure, is used in a supervised learning framework for link prediction. The data set used for experimental evaluation is based on co-authorship networks.

In the non-patent literature of Ryan N Lichtenwalter, Jake T Lussier, and Nitesh V Chawla, "New perspectives and methods in link prediction", in: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data

mining, pages 243-252. ACM, 2010, link prediction is done by supervised learning using a more systematic methodology both in the learning framework and in the evaluation matrix. The problem of an imbalanced sample set, i.e. sparsity of the social graphs is taken into account.

In the non-patent literature of Salvatore Scellato, Anastasios Noulas, and Cecilia Mascolo, "Exploiting place features in link prediction on local-based social networks", in: Proceedings of the 17th ACM SGKDD international conference on Knowledge discovery and data mining, pages 1046-1054, ACM, 2011 information is used about the location of users as additional features in the supervised learning setup to improve the classifier performance.

However one of the problems of conventional methods is that in many cases they take information other than the network structure into account leading to a huge amount of data. Further when the underlying network structure is not known detecting interactions can only be retrieved by sampling and the amount of samples that can be taken is restricted. Thus, conventional methods and systems provide a poor efficiency in detecting interactions.

An objective of at least one of the embodiments may be to provide a faster detection of interactions in an unknown network. A further objective of at least one embodiment may be to provide a higher success rate when discovering interactions without the necessity of using additional metadata.

At least one embodiment describes a method for detecting interactions between users of a network, performed in a memory available to one or more computation devices

comprising the steps of

a) Determining a maximum number of tests for detecting interactions
b) Performing a first number of tests by random sampling users for detecting interactions between them if any
c) Determining interactions between users by evaluating said first number of tests
d) Evaluating said determined interactions

e) Performing one or more further tests based on the prior evaluated interactions

f) Determining interactions between users using evaluations of said performed further tests,

g) Perform steps d)-g) again as as long as the maximum number of tests is not exceeded.

At least one embodiment describes a system for detecting interaction between users of network, comprising one or more computation devices, wherein a maximum number of tests for detecting interactions is predetermined, said computation devices comprising
first testing means adapted to perform a first number of tests by random sampling users for detecting interactions between them if any
first interaction means adapted to determine interactions between users by evaluating said first number of tests
evaluation means adapted to evaluate said determined interactions
second testing means adapted to perform one or more further tests based on the prior evaluated interactions
second interaction means adapted to determine interactions between users using evaluations of said performed further tests
and wherein said second means and said evaluation means are adapted such that their function being adapted to is only performed as long as the maximum number of tests is not exceeded.

In other words at least one embodiment describes a method and a system to faster detect interaction of mobile users with services or other mobile users. An application may be a backend application seeking to find as many such interactions as possible but information can also be retrieved by sampling that is applying an operation which receives information on whether a particular user or service interacts with a particular service or another user.

At least one of the embodiments may improve the success rate of interaction detection by taking into account the outcomes of previous samples without the necessity of additional metadata.

The term “service” is to be understood in the description and/or in the claims in a wide meaning: For example service may include but not limited to the services of a sensor, actuator or general internet of thing device or the like being also considered a service although there are substantial differences between service accessible of the internet or services provided by an internet of thing device.

The first and second testing means may be, but not limited to, an application adapted to perform testing running in a memory available to a processor, a microprocessor, a dual-core, quad-core or octa-core processor or the like or a processor with a memory. Said application or processor may have one or more interfaces, ports or the like for communication with other devices, entities, ports, interfaces or the like.

The first and second interaction means may be, but not limited to, an application adapted to perform interaction running in a memory available to a processor, a microprocessor, a dual-core, quad-core or octa-core processor or the like or a processor with a memory. Said application or processor may have one or more interfaces, ports or the like for communication with other devices, entities, ports, interfaces or the like.

The evaluation means may be, but not limited to, an application adapted to perform evaluation running in a memory available to a processor, a microprocessor, a dual-core, quad-core or octa-core processor or the like or a processor with a memory. Said application or processor may have one or more interfaces, ports or the like for communication with other devices, entities, ports, interfaces or the like.

Further features, advantages and further embodiments are described or may become apparent in the following:

In e) said prior evaluated interactions may be based on the first number of tests and already evaluated further tests. This may have the advantage of a continuous

enlargement of the number of prior evaluated interactions. Thus, efficiency and precision may be further enhanced.

For determining said further tests in e) a score may be computed representing a likelihood of interaction. This may provide a single value for example a real number to decide whether a further test has to be performed or not. This score may represent a relative likelihood, that there will be a link between the selected pair of users and/or services.

An undirected graph may be constructed representing already discovered interactions where nodes are represented by users and/or services. This undirected graph may be constructed such that the nodes represent the users and/or services while there may be an edge for each indirection between the respective user/user or user/service or service/service pair. This may enable an easy implementation and a fast detection of interactions.

For each node pair which has not been tested and for which no interaction has been found said score may be computed. This may avoid unnecessary resampling of already performed tests. Thus, efficiency may be enhanced.

The one or more highest scores may be used for performing a set of further tests. This may increase the probability of detecting interactions.

Additional metadata of users and/or services may be included when determining the first number of tests and/or said further tests. This may enhance the precision for detecting interactions even further.

Additional metadata of interactions may be included when determining the first number of tests and/or further tests. This may then further increase the precision for detecting interactions.

Said score may be calculated based on graph-based scoring functions. Said graph based scoring function may be at least one of the following procedure types or a combination of two ore more, e.g. by weighing the outcomes of said two or more

functions: Common neighbors, Adamic-Adar, Jaccard's coefficient or Preferential Attachment. These procedures are non-limiting examples of graph-based scoring functions. Of course other graph-based scoring functions can be used. For example common neighbors provide a scoring by counting a number of known neighbors to nodes of the graph have in common. Adamic-Adar uses a similarity measure for pairs of web pages based on features the pages have in common. When Jaccard's coefficient is used the number of common features is normalized by the total number of features both nodes have, and when using Preferential Attachment the scoring function is not based on measuring similarity of node pairs but only takes into account the individual probabilities of the nodes to have a link to a given other node.

Said score may be calculated with a statistical scoring function. This may enhance the precision and accuracy since statistics about the success of past tests are used. Of course also a combination of two ore more statistical scoring functions, e.g. by weighing the outcomes of said two or more functions can be used.

Said statistical scoring function may be of at least one of the following types: Average success, maximum success, success product. This may enhance the flexibility since success rates can be adapted according to the needs of a service operator or service provider. The success rate may be provided as the number of successes divided by the total number of tests applied. For nodes for which no adjacent edge has been tested yet the success rate may be set to a constant between 0 and 1.

For determining the further tests a supervised learning procedure may be used, wherein the evaluated first number of tests is used for training said supervised learning procedure. This may lead that interactions are discovered with a higher success rate since supervised learning can outperform unsupervised scoring functions for example. Since supervised learning methods are based on a classifier classifying data samples based on a vector of features where a number of pre-classified training samples are used for configuring the classifier beforehand the first number of tests may be used for training said supervised learning procedure in an easy and simple way.

For providing classifiers of said supervised learning procedure at least one of the following procedures may be used: NaïveBayes, MultiLayer Perceptron, Random Forest. This may provide classifiers in a reliable and fast way.

Said system may be a backend system comprising a user management entity comprising said first and second testing and said first and second interaction means and a link discovery entity comprising said evaluation means. This may enable an easy and efficient implementation.

There are several ways how to design and further develop the teaching of the embodiments mentioned above and further below in an advantageous way. To this end it is to be referred but not limited to the patent claims on the one hand and to the following explanation of embodiments of the invention by way of example, illustrated by the figure on the other hand. In connection with the explanation of the embodiments of the invention by the aid of the figure, generally embodiments and further developments of the teaching will be explained.

In the drawings

Fig. 1 shows a system according to a first embodiment;

Fig. 2 a configuration of a classifier for different methods;

Fig. 3 a success rate of graph-based scoring functions according to a further embodiment;

Fig. 4 a success rate of graph-based scoring functions for geographically distributed objects;

Fig. 5 a success rate of graph-based scoring functions on another data set and geographically distributed objects; and

Fig. 6 a success rate of graph-based scoring functions for a synthetic data set on geographically distributed moving objects according to an embodiment.

Fig. 1 shows a system according to a first embodiment.

In Fig. 1 a backend system BS is shown comprising a link discovery engine LDE and a user management entity UME. The user management entity UME samples mobile terminals MT for detecting interactions between the mobile terminals, i.e. the users. The user management entity UME then sends test samples to the link discovery engine LDE for evaluation. The user management entity UME may take multiple samples in parallel. The taken samples are then, as mentioned before, submitted to the link discovery engine LDE regardless of whether they are positive, i.e. there is an interaction, or negative, i.e. there is no interaction between user/service pairs or user/user pairs or service/service pairs.

The link discovery engine LDE then evaluates the received samples and provides pair ratings to the user management entity UME. The link discovery engine LDE updates its internal status and if the samples lead to a change in the ratings on the likelihood of interactions between the aforementioned pairs, the changes are communicated back to the user management entity UME. The link discovery engine LDE constructs a graph (V,E) representing already discovered interactions where a node set V comprises or represents the users and/or services while for each discovered interaction there is provided an edge in the edge set E. For each node pair that is not in the edge set E and that not has been sampled yet the link discovery engine LDE computes a score form the structure of the graph. This score represents a likelihood that an interaction, i.e. an edge is between the two nodes being sampled.

To predict links the future occurrence of edges given one or more snapshots of a social network has to be anticipated. What is considered here are graph-based scoring functions that take the neighborhoods of nodes into account for making predictions. These functions use a single snapshot of an undirected graph $G_0 = (V_0, E_0)$ as the input and return for each node pair $\{v_1, v_2\}$ not being an element of

E_0 a real number called the score. The score of a node pair represents the relative likelihood that there will be a link between them in a future version of the network.

To decide which edge to test next, scoring is applied to the partial graph defined by the full node set and all links discovered so far. Among all link candidates that have not been tested yet the one is selected that receives the highest score. The concrete graph-based scoring functions may be a subset of the ones evaluated in the non patent literature of David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. Journal of the American society for information science and technology, 58(7):1019-1031, 2007 for link prediction.

In the list below, $\Gamma(v)$ denotes the set of already discovered neighbors of node $v \in V$:

Common Neighbors as disclosed in the non patent literature of Mark EJ Newman. Clustering and preferential attachment in growing networks. Physical Review E, 64(2): 025102, 2011 is based on the following: The scoring is done by counting the number of known neighbors two nodes v_1, v_2 have in common, resulting in the formula

$$|\Gamma(v_1) \cap \Gamma(v_2)|.$$

Adamic-Adar as disclosed in the non patent literature of Lada A Adamic and Eytan Adar, Friends and neighbors on the web. Social networks, 25(3):211-230, 2003 is based on the following: Adamic-Adar proposes a similarity measure for pairs of web pages based on features the pages have in common. Interpreting the already discovered neighbors as the features as disclosed in the non patent literature of David Liben-Nowell and Jon Kleinberg, “The link-prediction problem for social networks”, Journal of the American society for information science and technology, 58(7):1019-1031, 2007, this results in the scoring function

$$\sum_{u \in \Gamma(v_1) \cap \Gamma(v_2)} \frac{1}{log|\Gamma(u)|}.$$

Jaccard's coefficient as disclosed in the non patent literature of David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. Journal of the American society for information science and technology, 58(7):1019-1031, 2007 is based on the following: In Jaccard's coefficient, another similarity measure, the number of common features is normalized by the total number of features both nodes have. Again interpreting neighbors as features, the resulting scoring function for node pair v_1, v_2 is

$$\frac{|\Gamma(v_1) \cap \Gamma(v_2)|}{|\Gamma(v_1) \cup \Gamma(v_2)|}.$$

Preferential Attachment as disclosed in the non patent literature of Albert-László Barabási and Réka Albert. Emergence of scaling in random networks, science, 286(5439):509-51, 1999 is based on the following: Here the scoring function is not based on measuring similarity of node pairs, but only taking into account the individual probabilities of the nodes to have a link to a given other node. In the popular network growth model of Barabási and Albert this probability is proportional to the number of links a node already has, and therefore the resulting scoring function is

$$|\Gamma(v_1)| \cdot |\Gamma(v_2)|.$$

Of course the results of these methods can be combined, e.g. by a weighted sum or the like.

The scoring functions for link prediction use knowledge gained from previous edge tests, and every edge test with positive result adds to that knowledge. Due to the sparsity of social networks, however, the vast majority of edge tests may have a negative result and therefore may not contribute to the knowledge these scoring functions can exploit. In order to also utilize knowledge from unsuccessful edge tests statistical scoring functions are used, as they use simple statistics about the success of past tests. For each node v of the number of successful and unsuccessful tests of edges {v,v'} adjacent to v is kept track. From these records the success rate r(v) is computed by dividing the number of successes by the total

number of tests applied. For nodes for which no adjacent edge has been tested yet the success rate, which would be undefined otherwise is set to a constant between 0 and 1.

Given the success rates of the individual nodes the score of any untested edge $\{v_1, v_2\}$ is computed as some function of $r(v_1)$ and $r(v_2)$. Concretely, the following functions can be used:

- *Average success*: $\frac{r(v_1)+r(v_2)}{2}$
- *Maximum success*: $\max\{r(v_1), r(v_2)\}$
- *Success product*: $r(v_1) \cdot r(v_2)$

Further supervised learning methods can be applied to link prediction, and such methods may outperform the unsupervised scoring functions. For applying supervised learning to link prediction, conventionally two snapshots of the graph are used, where the earlier snapshot is used for feature extraction, and the new edges that appear on the later one are used as training samples for the classifier, e.g. as disclosed in the non patent literature of Ryan N Lichtenwalter, Jake T Lussier, and Nitesh V Chawla. New perspectives and methods in link prediction. In Proceedings of the 16th ACM SGKDD international conference on Knowledge discovery and data mining, pages 243-252. ACM, 2010.

For making the following descriptions more intuitive the notion of time is used. Time is assumed to proceed in discrete steps t = 0,1,... and at each time step one edge test can be submitted. The first edge test may be executed at time 0; consequently at time t there are t test results available that can be used for determining the most promising candidate for the (t+1)-th test.

Supervised learning methods use so-called classifiers that classify data samples based on a vector of features, where a number of pre-classified training samples are used for configuring the classifier beforehand. In other words, each classifier has a training method and a classification method. The training method is called by using set of feature vectors with the corresponding classes as the input, while the classification method returns for a given feature vector the class it is likely to

belong to. The classifiers may also return a confidence value between zero and one. This value represents the confidence that the given node pair is comprised by the edge set of the social network.

The scoring functions defined previously can be used as features. For each candidate link the scores are computed using these functions and assign them as the components of the feature vector. The classifier is trained in regular time intervals, which is parameterized by the interval length k. In the first phase after time 0 the classifier may not be used for two reasons. Firstly, as no substantial number of edge tests has been applied yet, there are not enough pre-classified training samples. Secondly, also the computation of the features requires some amount of knowledge about the graph, so feature extraction does not give meaningful results in this early phase. Instead, random guessing in the first k time steps is applied. At time k the classifier for the first time can be run. A number of edges is picked that have been detected by tests and, to make the training set balanced, randomly the same number of node pairs is picked for which the edge test has been applied unsuccessfully. This sample set is submitted to the classifier for training. For each of these samples the feature vector at the time the sample is submitted to the classifier is not computed, but the feature values are used that were valid at the time the edge was tested. This improves the classifier accuracy.

After training, the classifier is ready for link discovery. For each node pair not tested yet the feature vector is computed and submitted to the classifier for classification, obtaining a confidence value. The k link candidates are selected with the highest confidence values and the edge tests are applied for them in time step k to 2k-1. After that, and every k-th time step from now on, the training of the classifier is repeated. To practically apply the method(s) just described, one may need to decide on four parameters. The first parameter is the training time interval k. The second parameter relates to the size of the training sample. When targeting balanced training samples like described above the size is restricted by the number of edges detected at the given point of time, but one can further upper bound the sample size by some constant. The third parameter is the classifier to be used, and finally the set of used features needs to be decided on. For reasons of computational efficiency the set of features may be restricted to a small number

of functions that are fast to compute. From the graph-based scoring functions Adamic-Adar and the Common Neighbors measure is chosen for the following. As for the statistical scoring functions, instead of using the different values derived from the success rates, the original numbers are used from which these scores are computed. For link candidate $\{v_1, v_2\}$, the four resulting features are the number of successful edge tests respectively involving v_1 and v_2, and the total number of tests of edges respectively having v_1 or v_2 as an endpoint.

The classifiers used in the experiments or in the following respectively are

- Naive Bayes as disclosed in the non patent literature of George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In Eleventh Conference on Uncertainty in Artificial Intelligence, pages 338-345, San Mateo, 1995. Morgan Kaufmann.,
- MultiLayer Perceptron as disclosed in the non patent literature of D. E. Rumelhart, G. E. Hinton, and R. J . Williams. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Learning Internal Representations by Error Propagation, pages 318-362. MIT Press, Cambridge, MA, USA, 1986., and
- Random Forest as disclosed in the non patent literature of Leo Breiman, Random forests. Mach. Learn., 45(1):5-32, October 2001.

In the following embodiment the capturing of the characteristics of a class of link discovery problems is described that appear in the context of mobile computing. Of course parts of the embodiment can be amended, replaced, and/or omitted.

A recommendation system is used which tracks the locations of its users. The system is expected to notify two users whenever they are close enough to each other and there is a social connection between them, for example because of common interests. The recommendation system is made aware of the location of users by receiving location updates, e.g. from GPS devices. For finding out whether two users are socially connected, the recommendation system uses an external service for applying edge tests. Answering these tests takes a non negligible amount of time, and during this time the external service cannot answer further requests.

Due to the time required for the edge tests, a model of discrete time can be assumed, where the discretization is done with respect to the time the edge test service needs for processing one user pair. Each time step may comprise three phases:

- In the update phase, a set of location updates arrives at the recommendation system, where every location update comprises of an identifier of a user and its current coordinates.
- In the testing phase, the recommendation system can choose one pair of users for applying the edge test, from which it receives a response stating whether or not the users are connected.
- In the third and final recommendation phase, the system assigns recommendations to a set of user pairs whose distance is currently less than a given threshold and has been certified to be connected.

Instances of this extended problem may be determined by a sequence of time-indexed location updates (encoding which ID/location pairs arrive for time step t : 1,. . .,T) and the binary matrix encoding which user pairs have overlapping interest. This scenario is an online problem as the instance is not completely known to the recommendation system in advance, and also future movements of users are not known to the recommendation system. The location updates become known from the time step they arrive, and whether two users match becomes known only after the recommendation system invests a time step to apply the edge test for this user pair. A recommendation of user pair (u_1,u_2) at time step t may be only feasible if

(a) the users are known to have a distance of d or less according to the location updates that have arrived by time t,

(b) an edge test applied at a time t or earlier has verified that the users are indeed connected, and

(c) the specific user pair has not been recommended at a previous time.

The recommendation system may be realized as follows: As the edge tests are assumed to take a certain amount of time, the time needed to determine the next

test to apply does not have to be close to real-time. Nevertheless, it has to be made sure that the time for the edge tests remains the bottleneck and no edge test opportunities are missed due to an overly long computation time of the process to decide on the next edge test. The interface exposed by implementations of the recommendation system may comprise of four functions, and in each time step these functions are subsequently called.

The first function may notifiy the recommendation system of updates regarding object movement. When this function is called, the recommendation system is expected to update its internal status and decide on the pair of users to apply the edge test for. This pair may then be retrieved by calling the second function, while the third function may be used to announce the result of the edge test to the recommendation system.

Finally, the fourth function may be called to retrieve the set of recommendations, i.e. user pairs that are geographically close and socially connected. Although the number of recommended user pairs cannot be more than one per time step on average, it is legitimate to apply edge tests regarding user pairs that are not geographically close and then wait until they become close before announcing the recommendation. This possibility is for example useful when all pairs of currently geographically close users have already been tested.

In the scenarios it is evaluated that there are enough close user pairs at nearly all times, so that there is no need to test edges between users not close to each other. In this embodiment at each time a user pair to test among all pairs that are close to each other and have not been tested yet, using one of the link discovery methods to make this selection. Although the implementation of the recommendation system depends on the specific link discovery method it implements, many parts can be realized in a generic way, which is described in the subsequent paragraphs. The recommendation system may maintain a number of data structures to keep track of the user locations, the edge tests it has already applied, and the respective answers.

In this embodiment, a spatial index to keep track of user positions, a test history hash table comprising the user pairs (edges) already tested and the respective answers, and a recommendation history table for the user pairs that have been recommended in the past are used. As auxiliary data structures to speed up computations, a location index may be additionally maintained as a hash map for quickly looking up the positions of given users, and a testing queue comprising the list of user pairs to test in the future, ordered by the score they receive from the link discovery method. Finally, a recommendation buffer comprising the user pairs that have been successfully tested but not yet recommended is maintained.

In the update phase of each time step, a number of user location updates arrives. The recommendation system updates the spatial index to account for the new user positions and updates the auxiliary location index as well. The location updates potentially cause new pairs of users that become geographically close to each other. These user pairs may be determined by a window query on the spatial index around the new position for each updated user. For each new pair of close users an operation on the test history reveals whether the corresponding edge has been tested in the past. If this is the case, and if the test was successful and the edge is not comprised by recommendation history, the edge is added to the recommendation buffer, so that it will be recommended in this time step. If the test history does not comprise the pair, it is added to the testing queue, where the priority of this pair is determined from the link discovery method in use.

In the testing phase of the time step, for deciding the concrete edge to test, the highest priority pair from the testing queue is selected. Regarding the maintenance of this queue, it is possible that the user pair is not mutually close anymore. The location index is used to look up the two user positions, and if they are not close to each other the pair is discarded and the next candidate from the testing queue is selected. This is done until the testing queue is either empty (in which case a random untested edge for testing is selected) or a user pair has been found that is still close. This edge may then be chosen to be tested.

When the response from the edge test is received in the testing phase of the time step, the result of that test is added to the test history. If the response is positive

and the users are geographically close, then the edge is added to the recommendation buffer to be recommended in this time step. If necessary, the specific data structures of the link discovery method used by the recommendation system are also updated with the results of the test. This can cause a change in the evaluation of the untested edges, and so the testing queue might have to be updated. In the computationally worst case each edge in the queue may get a new priority assigned, but for the methods aforementioned the number of updates is either low (for the scoring functions) or the updates are happening only in a small subset of all time steps (for the machine learning approach).

Finally, in the recommendation phase of the time step the recommendation buffer may be flushed and all edges contained by it are recommended.

To link the discovery methods two distinct data sets may be used. One data set represents a sample of a real social graph, whereas the other one has been generated by one of the most popular random models for social networks. The real data set has been obtained for the following from an Internet service called last.fm. Users of this service can listen to music, and a profile of their musical taste is automatically created. When users visit the profiles of other users they receive information on the compatibility of their musical tastes. So here the social network is defined by the users representing the nodes and the pairs of users with compatible musical taste are forming the edges.

The specific data set which is considered in the following has been obtained by applying random walks to the musical neighborhood graph of last.fm as shown in the non patent literature of Minas Gjoka, Carter T. Butts, Maciej Kurant, and Athina Markopoulou. Multigraph sampling of online social networks. IEEE JSAC on Measurement of Internet Topologies, 29(9), 2011. The resulting graph has about 400,000 users, where each user has an average number of 7.9 neighbors. In order to slightly increase the density of the graph and thus speed up the link discovery process of all methods, two iterations of removing all nodes of degree one have been performed. The final graph therefore contains about 264,000 users and the average degree is 10.9, where the maximum degree node has 60 neighbors.

The second data set has been generated by the random model described in the non patent literature of Riitta Toivonen, Jukka-Pekka Onnela, Jari Saramki, Jrkki Hyvnen, and Kimmo Kaski. A model for social networks. Physica A: Statistical Mechanics and its Applications, 371(2):851 – 860, 2006. To this end an existing generator called Nepidemix, available under http://nepidemix.irmacs.sfu.ca has been used, which is parameterized by the size N_0 of the seed network, the target size N of the network to be generated and the maximum number k of secondary nodes. From the seed network, which is generated as a ring of size N_0, the final network may be constructed by iteratively adding nodes. Whenever a new node is added, a small number of existing nodes are chosen as primary neighbors of the new node. From all nodes adjacent to the primary neighbors, a uniform random number between 0 and k of them may be chosen as secondary neighbors, where also this choice is uniformly at random. This procedure is repeated until the network has N nodes. Parameters being chosen for the generation are N_0=100, N=2000, and k=100. The resulting graph has 2000 nodes and an average degree of 47.5 with a maximum number of 173 neighbors, so it is considerably more dense than the real network.

To evaluate the link discovery methods for the moving object scenarios, additionally data representing object positions changing over time is required. A number of objects compatible with the respective number of nodes in the social graph under evaluation is generated. These objects are uniformly distributed over a geographic area that corresponds to the city of Heidelberg (about 1,4 km x 3,5 Km). The object movement was simulated in round-robin order, so that in each movement step some objects are moving by a distance of at most 200m into a random direction. Objects are defined to be close to each other (and thus eligible for link recommendations) when they have a distance of less than 400m (resp. 10m) to each other in case of the synthetic (resp. real) data set.

In the experimental setup for moving objects the situation is simulated that the edge tests require only little time compared to the time between two consecutive location updates of objects. The simulation environment has been configured such that between any two consecutive arrivals of location updates on the same object

the recommendation system can apply a large number of edge tests (20,000 in our experiments).

The machine learning methods used here rely on the functionality of the Weka framework as disclosed in the non patent literature of Weka, available under http://www.cs.waikato.ac.nz/ml/weka/, April 2014. The specific configurations of the Random Forest and Multilayer Perceptron classifiers can be found in Fig. 2.

In the following results of said experimental study are shown and plots on the same data sets use the same scaling factors in order to be comparable more easily.

First an experiment is performed where object locations are not taken into account and the synthetic data set has been used. The results of the experiment are shown in Figure 3 in form of curves that show how fast links become known over time. The first observation is that, compared to what is conventionally achieved in link prediction, the link discovery problem seems to be considerably more difficult. The highest improvement over random guessing that is achieved in this and all other experiments is a factor of less than 7. In this experiment it is observed that the link prediction scoring functions are outperforming both the statistical scoring functions and the more supervised learning methods. The latter achieve improvements over random guessing of factors less than 2.5. It is also apparent that the plots of Jaccard, Common Neighbors and Adamic-Adar are nearly exactly the same. This observation is consistent with all other experiments herein.

In the next experiment the objects are still not moving, but they are geographically distributed so that the set of object pairs for which links can be detected is smaller. The behavior of the link discovery methods is visualized in Figure 4. Three of the graph-based scoring functions are still outperforming the learning-based approaches, but when comparing the curves to Figure 3 it becomes apparent that the performance of the latter approaches is improving, while number of links the graph-based scoring functions can predict is slightly decreasing. It therefore seems that when the network becomes more sparse the machine learning methods improve their performance. In Figure 5 the results of the experiment with

full object movement are shown. Here the observations are similar to the previous experiment, but more pronounced. The best machine learning method, Random Forest, is outperforming random guessing by a factor of close to 4.5, while the factors of Jaccard, Common Neighbors and Adamic-Adar remain unchanged as compared to Figure 4.

In the final experiment the data set on last.fm users is used. As already mentioned previously, this graph is considerably more sparse and thus the links that are expected to be discovered with the same number of tests is accordingly smaller. On this sparse graph the success rates of all link discovery methods are substantially lower, and there are two that can outperform random guessing but the factor of 1.5. These are the machine learning methods Random Forest and Multilayer Perceptron. This observation again supports the interpretation that the supervised learning methods are e.g. better suited for more sparse networks.

To summarize at least one embodiment using supervised learning methods is capable to discover interactions more quickly than a pure chance. Here the Link Discovery Engine LDE starts with the random sampling phase and then extracts features from the links found so far, and additional uses some of these samples for training a classifier. The classifier is then used to compute the rating of node pairs. In the experiments described above the success rate of the sampling process could be improved by a factor of nearly 7 as compared to random sampling.

At least one embodiment does not depend on the availability of metadata about users like user profiles, services like semantic service descriptions or the interaction between both like time stamps, frequency of interaction, etc..

At least one embodiment can take into account this metadata when being available and said metadata can be taken into account by the link discovery engine potentially further increasing the success rate of the sampling process.

At least one embodiment has the advantage that the success rate of interaction detection is improved. This may be achieved by taking into account the outcomes of previous samples.

At least one embodiment comprises a method comprising the steps of an implementation of a concrete link discovering mechanism and an implementation of the aforementioned embodiments including a simple user management component and an application for the mobile terminals.

At least one embodiment enables that interactions are discovered with a higher success rate without the necessity of additional metadata.

At least one embodiment uses the outcomes of previous sampling steps in order to estimate the success likelihood of future samples. The user-service or user-user or service-service interactions can be represented by an undirected graph (V,E) where the node set (V) represents the users and services while there is an edge in E for each interaction between the respective user/service or user/user or service/service pair.

Due to restrictions like bandwidth computational resources, privacy or requirement-abstruseness the amount of samples that can be taken is restricted. In order to use the limited budget of samples as effectively as possible for detecting interactions at least one embodiment of the present invention gives preference to the most promising user-service or user-user or service-service candidate interaction pair when sampling.

At least one embodiment describes a method and a system for distinguishing and promising pairs from less promising pairs such that the success rate of the sampling process is substantially improved.

At least one embodiment can be generally applied for detecting all kinds of interactions like mobile users using public transport, mobile users being at the same events as other mobile users, mobile users taking advantage of e-coupons when shopping or users reacting to advertisement.

Further at least one embodiment does not depend on the availability of any domain-specific information on the users or the services. The improvement of the sampling success rate in comparison with random sampling may be achieved completely without the use of metadata. However, metadata can be taken into account of further increase the success rate. A mobile terminal or user equipment may be provided with an application exchanging information for interaction sampling with a user management entity.

Many modifications and other embodiments of the invention set forth herein will come to mind to the one skilled in the art to which the invention pertains having the benefit of the teachings presented in the foregoing description and the associated drawings. Therefore, it is to be understood that the invention is not to be limited to the specific embodiments disclosed and that modifications and other embodiments are intended to be included within the scope of the appended claims. Although specific terms are employed herein, they are used in a generic and descriptive sense only and not for purposes of limitation.

Claims

1. A method for detecting interactions between users of a network, performed in a memory available to one or more computation devices
comprising the steps of
   a) Determining a maximum number of tests for detecting interactions
   b) Performing a first number of tests by random sampling users for detecting interactions between them if any
   c) Determining interactions between users by evaluating said first number of tests
   d) Evaluating said determined interactions
   e) Performing one or more further tests based on the prior evaluated interactions
   f) Determining interactions between users using evaluations of said performed further tests
   g) Perform steps d)-g) again as long as the maximum number of tests is not exceeded.

2. The method according to claim 1, wherein in e) said prior evaluated interactions are based on the first number of tests and already evaluated further tests.

3. The method according to claim 1, wherein for determining said further tests in e) a score is computed representing a likelihood of interaction.

4. The method according to claim 1, wherein an undirected graph is constructed representing already discovered interactions, where nodes are represented by users and/or services.

5. The method according to claim 3 and 4, wherein for each node pair which has not been tested and for which no interaction has been found said score is computed.

6. The method according to claim 3, wherein the one or more highest scores are used for performing said further tests.

7. The method according to claim 1, wherein additional metadata of users and/or services is included when determining the first number of tests and/or further tests.

8. The method according to claim 1, wherein additional metadata of interactions is included when determining the first number of tests and/or further tests.

9. The method according to claim 3, wherein said score is calculated based on graph-based scoring functions.

10. The method according to claim 3, wherein said score is calculated with a statistical scoring function.

11. The method according to claim 10, wherein said statistical scoring function is of at least one of the following types: average success, maximum success, success product.

12. The method according to claim 1, wherein for determining the further tests a supervised learning procedure is used, wherein the evaluated first number of tests is used for training said supervised learning procedure.

13. The method according to claim 12, wherein for providing classifiers of said supervised learning procedure at least one of the following procedures are used: NaïveBayes, MultiLayer Perceptron, Random Forest.

14. A system for detecting interactions between users of a network, comprising one or more computation devices, wherein
a maximum number of tests for detecting interactions is predetermined, said computation devices comprising
first testing means adapted to perform a first number of tests by random sampling users for detecting interactions between them if any

first interaction means adapted to determine interactions between users by evaluating said first number of tests

evaluation means adapted to evaluate said determined interactions

second testing means adapted to perform one or more further tests based on the prior evaluated interactions

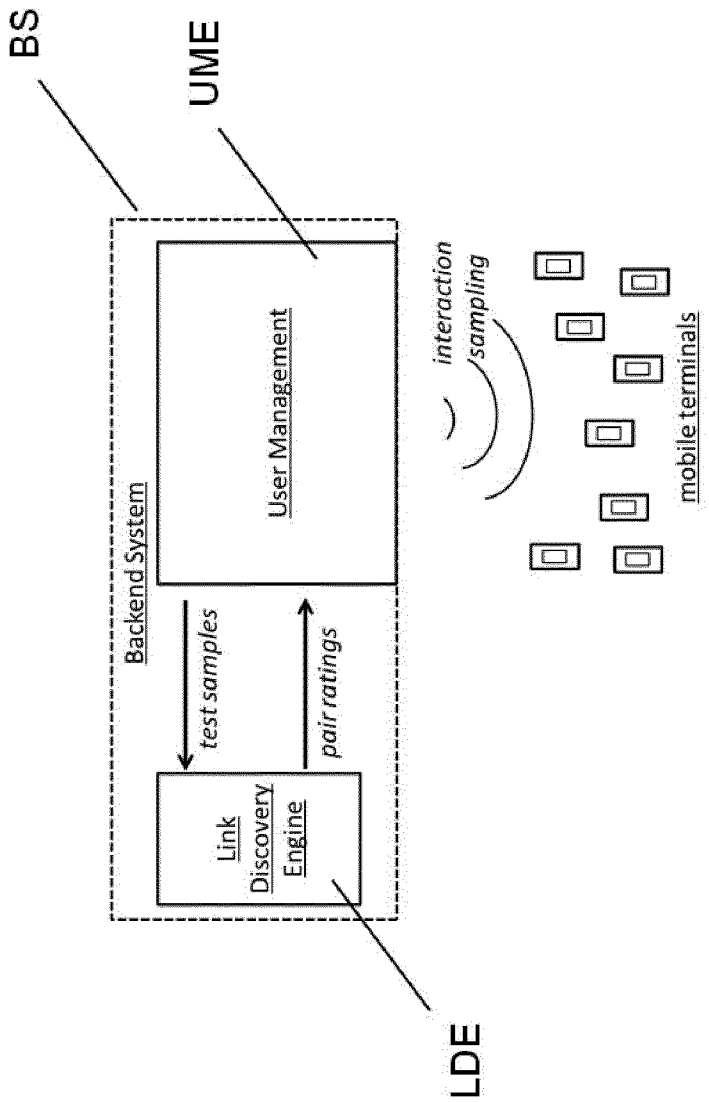
second interaction means adapted to determine interactions between users using evaluations of said performed further tests

and wherein said second means and said evaluation means are adapted such that their function being adapted to is only performed as long as the maximum number of tests is not exceeded.

15. The system of claim 14, wherein the system is a backend system comprising a user management entity comprising said first and second testing means and said first and second interaction means, and

a link discovery entity comprising said evaluation means.

Fig. 1


BS
UME
Backend System
User Management
test samples
pair ratings
Link Discovery Engine
LDE
interaction sampling
mobile terminals

Experiment	Random Forest	MultiLayer Perceptron
synthetic data	number of trees: 30 seed random generator: 1 depth: unlimited re-training interval: 5,000	momentum: 0.2 learning rate: 0.5 number of epochs: 30 size of validation set: 0 seed: 0 max consecutive errors: 20 hidden layers: 7 re-training interval: 5,000
synthetic data, distributed objects	number of trees: 20 seed random generator: 1 depth: unlimited re-training interval: 5,000	momentum: 0.2 learning rate: 0.5 number of epochs: 30 size of validation set: 0 seed: 0 max consecutive errors: 20 hidden layers: 7 re-training interval: 5,000
synthetic data, moving objects	number of trees: 20 seed random generator: 1 depth: unlimited re-training interval: 5,000	momentum: 0.2 learning rate: 0.5 number of epochs: 50 size of validation set: 0 seed: 0 max consecutive errors: 20 hidden layers: 7 re-training interval: 5,000
last.fm data	number of trees: 15 seed random generator: 1 depth: unlimited re-training interval: 1.6 M	momentum: 0.2 learning rate: 0.5 number of epochs: 150 size of validation set: 0 seed: 0 max consecutive errors: 20 hidden layers: 4 re-training interval: 1.6 M

Fig. 2

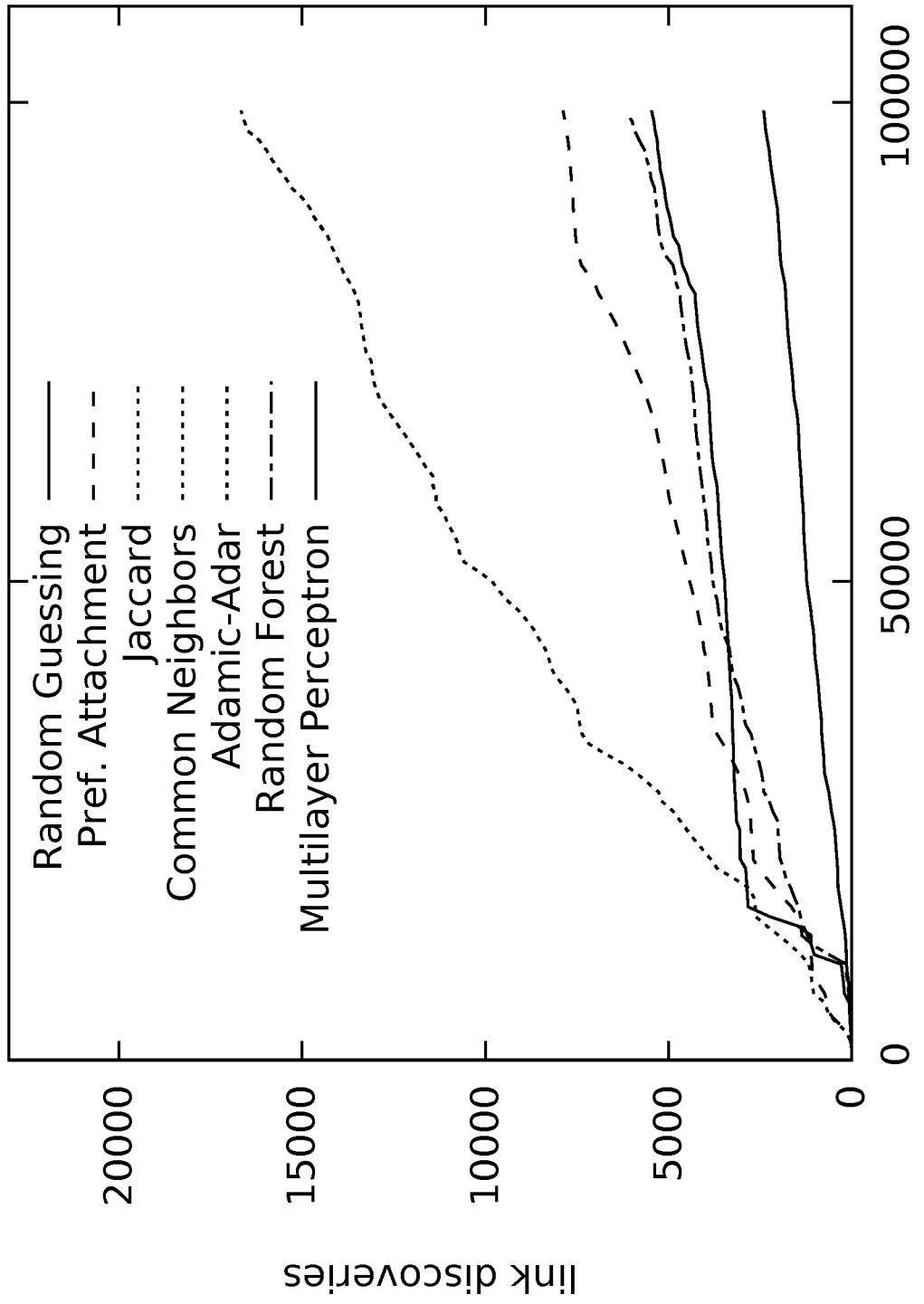

Random Guessing
Pref. Attachment
Jaccard
Common Neighbors
Adamic-Adar
Random Forest
Multilayer Perceptron
link discoveries
edge tests
0
5000
10000
15000
20000
0
50000
100000

Fig. 3

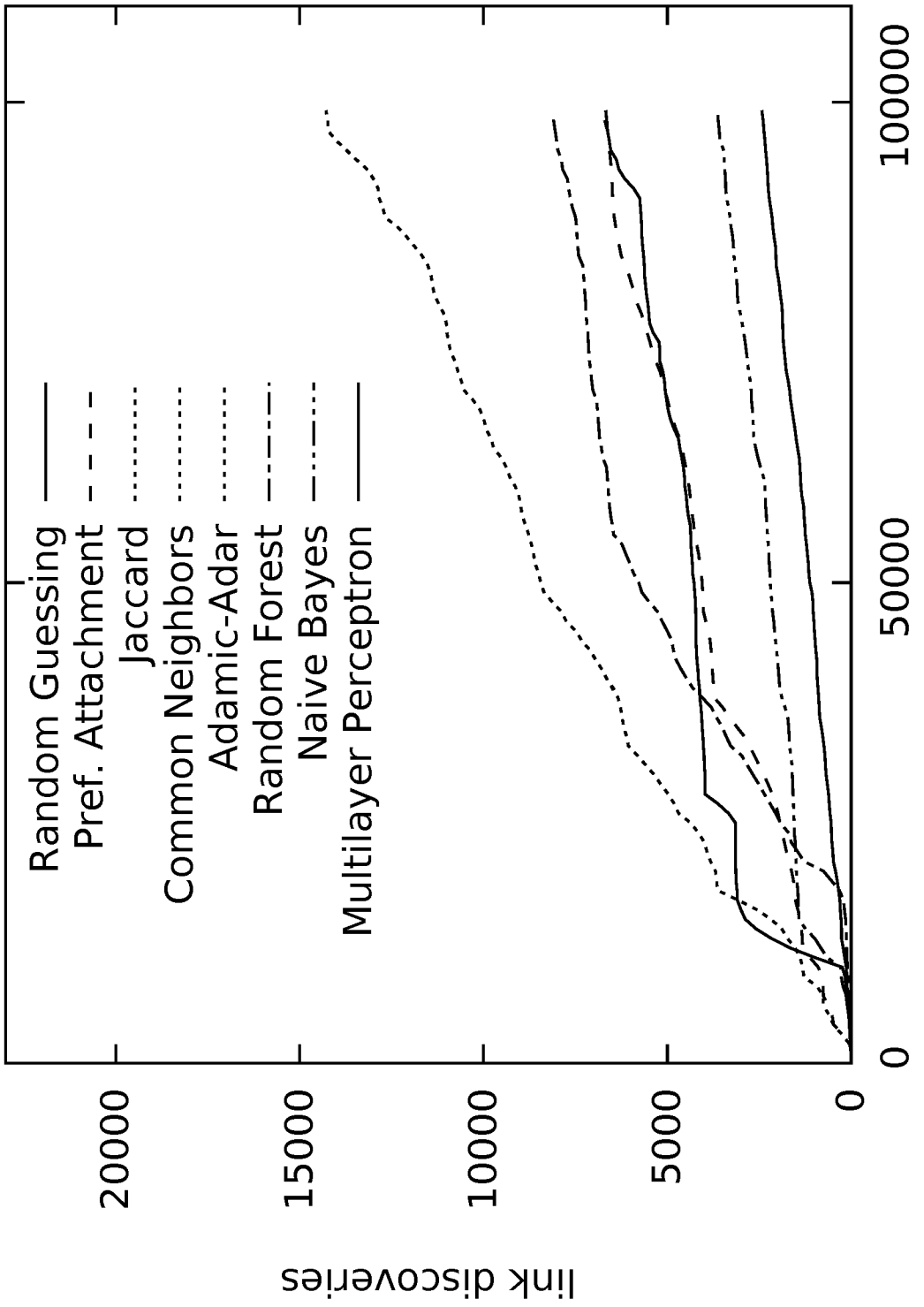

Random Guessing
Pref. Attachment
Jaccard
Common Neighbors
Adamic-Adar
Random Forest
Naive Bayes
Multilayer Perceptron
link discoveries
20000
15000
10000
5000
0
0
50000
100000
edge tests

Fig. 4

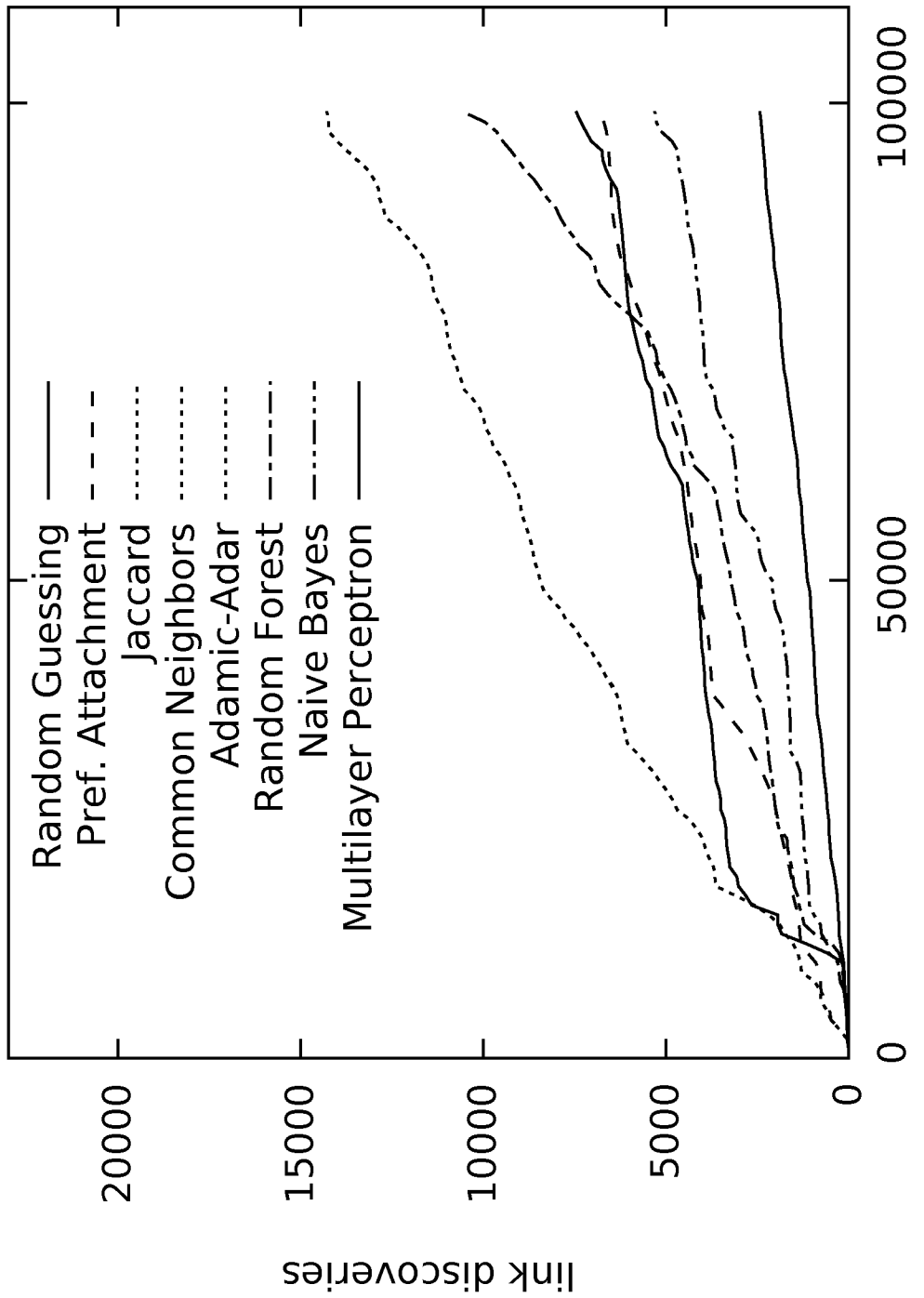

Random Guessing
Pref. Attachment
Jaccard
Common Neighbors
Adamic-Adar
Random Forest
Naive Bayes
Multilayer Perceptron
link discoveries
edge tests
0
5000
10000
15000
20000
0
50000
100000

Fig. 5

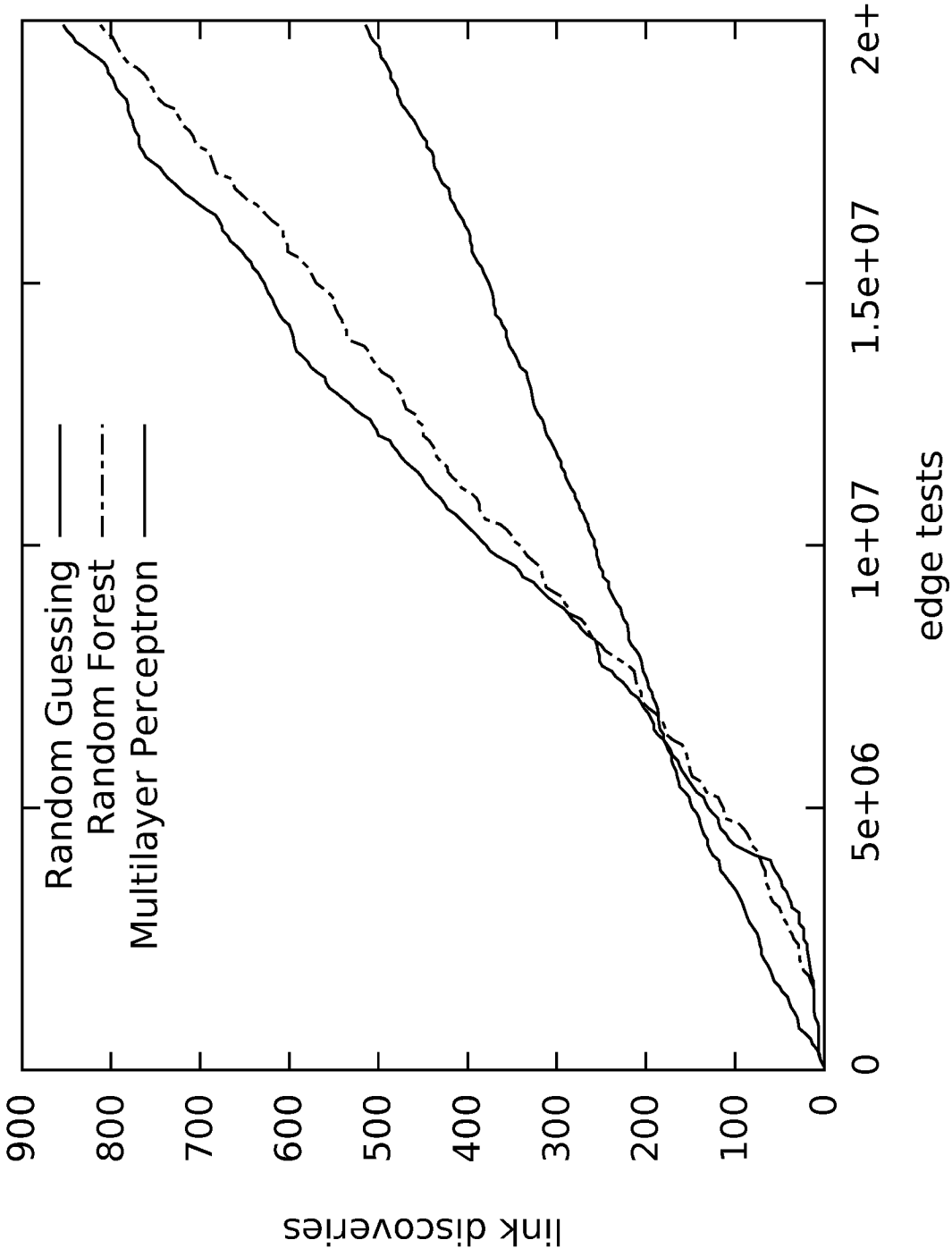

link discoveries
edge tests
Random Guessing
Random Forest
Multilayer Perceptron
0
100
200
300
400
500
600
700
800
900
0
5e+06
1e+07
1.5e+07
2e+07

Fig. 6

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2015/059030

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06Q50/00
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06Q

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
	The claimed subject-matter, with due regard to the description and drawings in accordance with Rule 33.3 PCT, relates to processes comprised in the list of subject-matter and activities for which no search is required under Rule 39 PCT. The information technology employed as an enabler for carrying out said processes is conventional. Its use for carrying out non-technical processes forms part of common general knowledge and it was widely available to everyone at the date of filing of the present application. No documentary evidence is therefore considered necessary. (See Official Journal EPO 11/2007, pages 592ff and 594ff) -----	

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
23 July 2015	31/07/2015

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	van Praagh, Kay