Simple User Assistance by Data Posting (Discussion Paper)

Elio Masciari, Domenico Saccà, and Irina Trubitsyna
elio.masciari@icar.cnr.it
{sacca,trubitsyna}@dimes.unical.it

ICAR-CNR, DIMES-Università della Calabria, 87036 Rende (CS), Italy

Abstract. Big Data deeply changed both researchers and industries approaches to data management. This impressive amount of data contains a lot of information regarding user suggestions and searches. In this paper, we focus on the effect of users suggestions on their social environment. Suggestions are provided to user by means of simple rules that are defined on the basis of the data being analyzed. We address the reaction of users in a competitive environment when they were invited to judge each other choices. The above mentioned activity is crucial to perform efficient identification of users able to spread their influence across the network, which is a key activity in several scenarios like tourism promotion, personalized marketing, and entertainment suggestion.

1 Introduction

Big Data paradigm is currently driving both research and industrial interests. Many proposals for innovative data analysis for managing impressive amounts of data have been developed. These approaches share the common needs to rethink storage and analysis techniques in order to deal with massive, heterogeneous data, that are fast varying [1]. Indeed, the Big Data revolution has been guided by the continuous advances of the web-based services that have millions of users generating new data everyday. As an example, applications such as Uber, Facebook or Twitter attract millions of people posting information about their habits and activities. Quick analysis of these data is a key activity in several scenarios, such as tourism promotion, personalized marketing and entertainment offers.

To this end in [6] a collaborative network that allows users to cooperate each others for high performance task execution (by partitioning complex tasks in smaller and easier sub-tasks) is described. Each user can operate as client and/or executor. As the tasks assigned to executors may consist in processes defining specific features for a context (e.g., deciding whether a restaurant is better than another), it may happen to deal with inconsistent or incomplete data. Tasks requiring details of all possible solutions obtainable with data exchange process can be solved applying approximate certain query answering algorithms [8–10, 12]. To extract concise data chosen among the possible alternatives on the basis of analyst's indications the data posting framework can be used [2]. The latter formalism allows us to model different situations. However, it requires the ability to manage *count constraints* and *domain relations* (we formally describe these notions in the background section). To simplify the modelling process, we propose to integrate the standard mapping with some parameters, avoiding complex specifications. We call the obtained rules *smart mapping rules* (more details can be found in [13, 14]. As an example, in Section 3 we consider the task of generating user suggestions on the basis of information posted by other peoples and available from the Web. We show how this process can be modelled with smart mapping rules.

Since the sub-tasks are solved by independent executors, payed for their activities, an important problem regards the quality evaluation of the obtained results. In our context the clients can ask for search directions. Once the result is obtained, we can ask the client to evaluate the obtained suggestion, i.e. give a rank for the executor performing this task. Moreover, some credits to executors suggesting useful search directions to clients are assigned. We analyze what happens with users' evaluations when they are aware of the other ones. In particular, we consider a group of users involved in a research project and we let them know mutual preferences on some topics.

We use Exponential Graph Random Models (EGRM) [3] to analyze the network dynamics and to model the revenue flow generated by user interactions. Since the maximum likelihood evaluation is computationally hard, we use two approaches for sampling data: *Metropolis Hastings sampling* and *Clustering based sampling*. Moreover, since the data are continuously updated, we use two type of statistical analysis: *cross-sectional* and *dynamic*.

It has been shown in literature, that in many settings the possibility to observe the other user rankings causes mutual adjustment among raters. In our framework, higher rankings are associated with positive evaluation (thus higher rewards), such that being ranked below others is aversive. To properly measure the above mentioned network dynamics, we compute three values proposed in [11]: (*i*) *Global Nonconformity (GNC)* - indicates how user conforms with others'; (*ii*) *Local Nonconformity (LNC)* - indicates how user conforms with neighbours; (*iii*) *Deference Aversion (DA)* - measures the following phenomenon: if user A ranks B above C ($B > C$), and user C would like rank A above B ($A > B$), since the transitivity of ranking would state that A is above C, user C is influenced negatively to express his ranking as above defined. Based on these measures we performed a deep experimental analysis.

2 Preliminaries

The data posting framework [7] adapts the well-known data exchange techniques to the new Big Data management and analysis challenges we find in real world scenarios. The *data posting setting* $(\mathbf{S}, \boldsymbol{\mathcal{D}}, \mathbf{T}, \Sigma_{st}, \Sigma_t)$ consists of a finite source database schema $\mathbf{S}$, a finite domain database scheme $\boldsymbol{\mathcal{D}}$, a target schema $\mathbf{T}$, a set Σ_{st} of non-deterministic source-to-target TGDs and a set Σ_t of target count constraints. A *non-deterministic source-to-target TGD (NdTGD)* is a dependency over $\langle \mathbf{S}, \boldsymbol{\mathcal{D}}, \mathbf{T} \rangle$ of the form $\forall \mathbf{x} \, [\, \phi_S(\mathbf{x} \cup \tilde{\mathbf{y}}) \rightarrow \phi_T(\mathbf{z}) \,]$, where $\mathbf{x}$ and $\mathbf{z}$ are lists of universally quantified variables; $\tilde{\mathbf{y}}$ is a (possibly empty) list of variables, called *non deterministic*, these variables can occur in ϕ_S only in relations from $\boldsymbol{\mathcal{D}}$; $\mathbf{x} \cap \tilde{\mathbf{y}} = \emptyset$ and $\mathbf{z} \subseteq \mathbf{x} \cup \tilde{\mathbf{y}}$; the formula ϕ_S and ψ_T are conjunctions of atoms with predicate symbols in $\mathbf{S} \cup \boldsymbol{\mathcal{D}}$ and in $\mathbf{T}$, respectively. The structure of NdTGDs ensures that any target database on $\mathbf{T}$ is finite. The NdTGD can be seen as the standard TGD, where existentially quantified variables are replaced with non-deterministic variables, whose values can be chosen from the finite domains defined by domain relations, i.e. relations from $\boldsymbol{\mathcal{D}}$. The mapping process is performed as usual but presumes that for every assignment of $\mathbf{x}$ a subset of all admissible values for $\tilde{\mathbf{y}}$ can be chosen in an arbitrary way. A *count constraint* is a dependency over $\mathbf{T}$ of the form $\forall \mathbf{x} \, [\, \phi_T(\mathbf{x}) \rightarrow \#(\{\mathbf{Y} : \exists \mathbf{z} \, \alpha(\mathbf{x}, \mathbf{Y}, \mathbf{z})\})$ `<op>` $\beta(\mathbf{x}) \,]$, where ϕ_T is a conjunction of atoms with predicate symbol in $\mathbf{T}$, `<op>` is any of the comparison operators ($=$, $>$, $\geq$, $<$ and $\leq$), $H = \{\mathbf{Y} : \exists \mathbf{z} \, \alpha(\mathbf{x}, \mathbf{Y}, \mathbf{z})\}$ is a *set term*, $\#$ is an interpreted function symbol that computes the cardinality of the (possibly empty) set corresponding to H, $\#(H)$ is a *count term*, and $\beta(\mathbf{x})$ is an integer or a variable in $\mathbf{x}$ or another count term with universally quantified variables in $\mathbf{x}$. The lists $\mathbf{x}$, $\mathbf{Y}$ and $\mathbf{z}$ do not share variables, $\alpha(\mathbf{x}, \mathbf{Y}, \mathbf{z})$ is a conjunction of atoms $T_i(\mathbf{x}, \mathbf{Y}, \mathbf{z})$ with $T_i \in \mathbf{T}$. Let I_T be the instance of $\mathbf{T}$. The *active domain* AD_I is the set of all values occurring in I_T. Given a substitution $\mathbf{x}/\mathbf{v}$ assigning values in AD_I to universally quantified variables, $K_{\mathbf{v}} = \{\mathbf{Y} : \exists \mathbf{z} \, \alpha(\mathbf{v}, \mathbf{Y}, \mathbf{z})\}$ defines the set of values in AD_I assigned to the free variables in $\mathbf{Y}$ for which $\exists \mathbf{z} \, \alpha(\mathbf{v}, \mathbf{Y}, \mathbf{z})$ is satisfied by I_T and $\#(K_{\mathbf{v}})$ is the cardinality of this set. We say that I_T satisfies the count constraint if each substitution $\mathbf{x}/\mathbf{v}$ that makes true its body expression, makes also true its head expression. The *data posting problem* is defined as follows: given finite source instance I_S and finite domain instance $I_{\mathcal{D}}$, find a finite target instance I_T such that $\langle I_S, I_{\mathcal{D}}, I_T \rangle$ satisfies both Σ_{st} and Σ_t. This problem is $\mathcal{NP}$-complete under the data complexity. When Σ_{st} does not contain non-deterministic variables, the problem becomes polynomial.

3 Setting up user suggestions

In this section we consider the task requiring generation user suggestions on the basis of information posted in the Web. In the following example we show how this process can be modelled with smart mapping rules.

Example 1. Suppose to have the description of the user ratings (that we call comments) stored in the relation $\mathtt{C(U,N,V)}$ with attributes $\mathtt{U}$ (user identifier), $\mathtt{N}$ (argument of rating) and $\mathtt{V}$ (rating value). Suppose also to have a relation $\mathtt{T(U_1,U_2,L)}$ that represents the trust level $\mathtt{L}$ of user $\mathtt{U_2}$ for the user $\mathtt{U_1}$.

In order to suggest to the user some "relevant" arguments we can set the following criterion: *"An argument n is suggested to the user u if the following* *conditions hold: 1) it is* sufficiently supported*, i.e. it is "supported" by at least* 20 *users whose trust level towards u is greater than* 70%*; 2) if different comments corresponding to the same argument are sufficiently supported, only one can be selected, with the greater level of support".* In this case, the choice of the comment has to take into account two different needs: 1) deciding if the argument has to be suggested to the user; 2) selecting the rating value for this argument.

This scenario can be modeled as follows:

- C and T are source relations;
- Add(U, N, V) is target relation, specifying the comments to be suggested to U;
- the smart mapping rule is reported below:

$$\mathtt{C}(\mathtt{u_2}, \mathtt{n}, \mathtt{v}) \wedge \mathtt{T}(\mathtt{u_1}, \mathtt{u_2}, \mathtt{l}) \wedge \mathtt{l} > \mathtt{0,70} \xrightarrow{\mathtt{u_2}, \mathtt{20}, \langle \mathtt{v}, \mathtt{unique} \wedge \mathtt{max} \rangle} \mathtt{Add}(\mathtt{u_1}, \mathtt{n}, \mathtt{v})$$

Intuitively, the body of the rule allows us to restrict the attention to the users whose trust level towards $\mathtt{u_1}$ is greater than 70%, the selection criterion has been synthesized on the arrow, indicating 1) the support variable $\mathtt{u_2}$, 2) the minimum quantity 20 of support instances to be able to map, and 3) the variable v whose value should be chosen from the set of candidate values following the indication $\mathtt{unique} \wedge \mathtt{max}$. □

More formally, a *smart mapping rule* has the form $\forall \mathbf{z} [\, \phi_S(\mathbf{z}) \xrightarrow{\mathbf{y}, k, \langle \mathbf{v}, f \rangle} r(\mathbf{x}, \mathbf{v}) \,]$, where $\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{v}$ are vectors of variables, such that $\mathbf{x} \cup \mathbf{y} \cup \mathbf{v} \subseteq \mathbf{z}$ and $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{v}$ do not share the variables; ϕ_S is the conjunction of source relations and expressions involving comparison operators ($>, <, \geq, \leq, =, \neq$) and variables in $\mathbf{z}$ or constants; r is a target relation; $\mathbf{y}$ is called a *support vector*; k is a natural number (greater than 0) which indicates the support value; the pair $\langle \mathbf{v}, f \rangle$ indicates how the choice for the values of $\mathbf{v}$ should be performed: f can be *max*, *unique*, the conjunction $unique \wedge max$, or empty.

We assume that each target relation can be defined by only one smart mapping rule. The smart mapping rule specifies that the tuple $\langle \mathbf{X}, \mathbf{V} \rangle$ is added to r only if it is supported by at least k (different) initializations $\{\mathbf{Y_1}, ...\mathbf{Y_k}\}$ of $\mathbf{y}$, i.e. for each $j \in [1..k]$ there exists an initialization $\mathbf{Z_j}$ of $\mathbf{z}$, that maps $\mathbf{x}$, $\mathbf{y}$ e $\mathbf{v}$ in $\mathbf{X}$, $\mathbf{Y_j}$ and $\mathbf{V}$ respectively, and that makes true $\psi_S(\mathbf{Z_j})$. In the case than no further indications of choice are specified (the third arrow label is $\langle \mathbf{v}, \rangle$) all the tuples supported by at least k (different) initializations of $\mathbf{y}$ are added to r. Otherwise, the set of tuples to be added is further reduced using f content for the selection of values in $\mathbf{v}$.

The $\langle \mathbf{v}, unique \rangle$ constraints specifies that the r relation must obey the functional dependency $\mathbf{x} \rightarrow \mathbf{v}$, i.e. for each assignment of values in $\mathbf{x}$ the assignment of values in $\mathbf{v}$ must be unique. In the case that several tuples are supported by at least k (different) initializations of $\mathbf{y}$ and they have the same values in $\mathbf{x}$, the choice can be made arbitrarily.

The indication $\langle \mathbf{v}, max \rangle$ specifies that, for each $\mathbf{X}$ only tuples supported by a maximum number of initializations of $\mathbf{y}$ must be selected. It is easy to see that this indication does not guarantee the uniqueness of the choice. For example, it is possible that two tuples $\langle \mathbf{X}, \mathbf{V_1} \rangle$ and $\langle \mathbf{X}, \mathbf{V_2} \rangle$ have the same degree of support corresponding to the maximum value.

The indication $\langle \mathbf{v}, \mathit{unique} \wedge \mathit{max} \rangle$ specifies that, fixed $\mathbf{X}$, only one tuple $\langle \mathbf{X}, \mathbf{V} \rangle$ can be chosen among those supported by a maximum number of (different) initializations of $\mathbf{y}$.

The set of smart mapping rule can be translated into the standard data posting constructs by translating every mapping rule ρ as follows. We introduce the unary domain relation $\mathcal{D}_\rho$. If f is empty $\mathcal{D}_\rho$ contains values -1 and 1, otherwise $\mathcal{D}_\rho$ contains values -1, 0 and 1. We also introduce the target relations $A_\rho(\mathbf{X}, \mathbf{Y}, \mathbf{V})$ and $Add_\rho(\mathbf{X}, \mathbf{V}, Flag)$, where $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{V}$ represent vectors of attributes corresponding to the vectors of variables $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{v}$, respectively, while the decision weather to select the pair $\langle \mathbf{x}, \mathbf{v} \rangle$ in the target relation r is stored by the attribute $Flag$: -1 or 0 (not added) and 1 (added). The mapping rules are:

$$\phi_S(\mathbf{z}) \rightarrow \mathtt{A}_\rho(\mathbf{x}, \mathbf{y}, \mathbf{v})$$
$$\phi_S(\mathbf{z}) \wedge \mathcal{D}_\rho(\mathtt{flag}) \rightarrow \mathtt{Add}_\rho(\mathbf{x}, \mathbf{y}, \mathtt{flag})$$

where all the variables are universally quantified. Using the domain relation $\mathcal{D}_\rho$ ensures that only a value among -1, 0 and 1 can be chosen for each initialization of $(\mathbf{x}, \mathbf{y})$ in the relation $\mathtt{Add}_\rho$.

The set of target count constraints is constructed as follows.

1. We start by adding *s*upport constraints, that ensure that the value -1 is assigned to the variable `flag` iff the degree of support of the combination $\langle \mathbf{x}, \mathbf{v} \rangle$ does not reach k. First, we add constraint for values 1 and -1.

$$\mathtt{Add}_\rho(\mathbf{x}, \mathbf{v}, 1) \longrightarrow \#(\{\, \mathbf{Y} : \mathtt{A}_\rho(\mathbf{x}, \mathbf{Y}, \mathbf{v})\}) \geq \mathtt{k}$$
$$\mathtt{Add}_\rho(\mathbf{x}, \mathbf{v}, -1) \longrightarrow \#(\{\, \mathbf{Y} : \mathtt{A}_\rho(\mathbf{x}, \mathbf{Y}, \mathbf{v})\}) < \mathtt{k}$$

If f is not empty, we also add constraint for value 0:

$$\mathtt{Add}_\rho(\mathbf{x}, \mathbf{v}, 0) \longrightarrow \#(\{\, \mathbf{Y} : \mathtt{A}_\rho(\mathbf{x}, \mathbf{Y}, \mathbf{v})\}) \geq \mathtt{k}$$

2. When $f = \mathit{unique}$ the *uniqueness choice constraint* is added:

$$\mathtt{Add}_\rho(\mathbf{x}, \_, \mathtt{flag}) \wedge \mathtt{flag} \geq 0 \rightarrow \#(\{\, \mathbf{V} : \mathtt{Add}_\rho(\mathbf{x}, \mathbf{V}, \mathbf{1})\}) = 1$$

This constraint ensures that exactly one initialization of $\mathbf{v}$ for each $\mathbf{X}$ is selected.

3. When $f = \mathit{max}$ we add (i) the *optimization constraint*

$$\mathtt{Add}_\rho(\mathbf{x}, \mathbf{v_2}, 1), \mathtt{Add}_\rho(\mathbf{x}, \mathbf{v}, 0) \longrightarrow \#(\{\mathbf{Y} : \mathtt{A}_\rho(\mathbf{x}, \mathbf{Y}, \mathbf{v_2})\}) \geq \#(\{\mathbf{Y} : \mathtt{A}_\rho(\mathbf{x}, \mathbf{Y}, \mathbf{v})\})$$

(ii) the *choice constraint*, ensuring that at least one initialization of $\mathbf{v}$ for each $\mathbf{X}$ is selected

$$\mathtt{Add}_\rho(\mathbf{x}, \_, \mathtt{flag}) \wedge \mathtt{flag} \geq 0 \rightarrow \#(\{\, \mathbf{V} : \mathtt{Add}_\rho(\mathbf{x}, \mathbf{V}, \mathbf{1})\}) \geq 1$$

4. When $f = \mathit{unique} \wedge \mathit{max}$ we add the uniqueness choice constraint and the optimization constraint.

The use of the smart mapping rules offers a notable simplification of the data posting framework. The user and/or designer can construct the selection criterion focusing only on the a small set of parameters. The selection criteria modeled by smart mapping rules are the most widespread in practice. The standardization of their representation allows us also to optimize the implementation of the data posting process. Smart mapping rules, initially developed for our application scenario, can be profitable used in different contexts. In particular, we are investigating their application in P2P Deductive Databases [4, 5].

4 Experimental Evaluation

In this section we describe the evaluation of the user ranking behaviours. We report the results obtained on a test set of user searches collected for four months during a research project. We devised for the experimental evaluation an approach similar to the one presented in [2] where the authors leverage ERGM and clustering analysis to partition the user interaction graph in order to tackle the complexity issues due to graph structures.

We provided each user with two different options: 1) they can ask for search suggestion or 2) they can suggest search directions on request. At fixed time intervals, users may evaluate obtained results (both as consumer and provider of information) by assigning a rank ranging from 0 to 10 to their experience. As we implemented a rewarding strategy, i.e. users pay for targeted suggestions and are paid when they suggest a proper search, the effectiveness evaluation is crucial as we do not want user to pay for wrong information.

We model our small network interactions by a random graph that is continuously updated as new links among users appear (i.e. new interactions between a pair of users). In order to better understand the information relevant to our goal we leverage two types of analysis: the first one, referred as cross-sectional (CS), considers only the variation occurring when data are observed while the second one considers each observation as an independent unit, thus it is referred as dynamic (Dyn). In order to be fair and to evaluate the influence excerpted by users each other, every participant is aware of the other's scores.

The latter results in the need to analyze factors that are *endogenous*, i.e., those phenomenon that may arise when users know each the rankings of other thus causing the rankings given by user i to be influenced by other user rankings except i. To this end we measure the values of GNC, LNC and DA.

Evaluation. In order to analyze the evolution of user search effects, we use the following statistics: overall searches number performed by user; keyword total number; mean assigned keyword rank. For the sake of completeness, we use both CS and Dyn analysis combined with clustering (CL) and Metropolis-Hastings (M-H) sampling. The results are shown in Figure 1. For Dyn analysis we compute fifteen graphs (one for each week except the first one) for the CS analysis we have 16 graphs. It is easy to observe that in tables a and b the values obtained for DA are always high since the beginning. It is worth noting that, as we start

Week	*GNC*	*LNC*	*DA*
1	0.000	−0.012	−0.127
2	0.001	−0.014	−0.131
3	0.001	−0.015	−0.127
4	0.001	−0.021	−0.144
5	0.002	−0.024	−0.198
6	0.002	−0.037	−0.223
7	0.003	−0.043	−0.289
8	0.003	−0.039	−0.311
9	0.002	−0.038	−0.327
10	0.0015	−0.041	−0.381
11	0.0015	−0.037	−0.401
12	0.001	−0.035	−0.412
13	0.001	−0.033	−0.423
14	0.001	−0.032	−0.428
15	0.000	−0.030	−0.436
16	0.001	−0.029	−0.441

a) CS, M-H

Week	*GNC*	*LNC*	*DA*
1 → 2	0.000	−0.009	−0.185
2 → 3	0.001	−0.011	−0.190
3 → 4	0.001	−0.015	−0.195
4 → 5	0.001	−0.020	−0.257
5 → 6	0.001	−0.022	−0.289
6 → 7	0.002	−0.023	−0.303
7 → 8	0.002	−0.026	−0.313
8 → 9	0.002	−0.026	−0.357
9 → 10	0.002	−0.029	−0.383
10 → 11	0.001	−0.030	−0.398
11 → 12	0.001	−0.031	−0.401
12 → 13	0.001	−0.031	−0.409
13 → 14	0.001	−0.032	−0.421
14 → 15	0.001	−0.033	−0.435
15 → 16	0.001	−0.033	−0.486

b) Dyn, M-H

Week	*GNC*	*LNC*	*DA*
1	0.000	−0.011	−0.111
2	0.001	−0.012	−0.122
3	0.001	−0.018	−0.1143
4	0.001	−0.020	−0.154
5	0.002	−0.021	−0.161
6	0.003	−0.042	−0.173
7	0.002	−0.048	−0.174
8	0.001	−0.052	−0.176
9	0.001	−0.060	−0.179
10	0.001	−0.061	−0.180
11	0.001	−0.068	−0.182
12	0.001	−0.069	−0.183
13	0.001	−0.071	−0.184
14	0.001	−0.073	−0.184
15	0.001	−0.075	−0.184
16	0.001	−0.079	−0.185

c) CS, CL

Week	*GNC*	*LNC*	*DA*
1 → 2	0.000	−0.008	−0.101
2 → 3	0.000	−0.011	−0.119
3 → 4	0.000	−0.013	−0.127
4 → 5	0.001	−0.019	−0.138
5 → 6	0.000	−0.021	−0.145
6 → 7	0.000	−0.028	−0.167
7 → 8	0.001	−0.031	−0.172
8 → 9	0.000	−0.032	−0.169
9 → 10	0.000	−0.037	−0.166
10 → 11	0.000	−0.041	−0.152
11 → 12	0.001	−0.044	−0.148
12 → 13	0.000	−0.053	−0.145
13 → 14	0.000	−0.057	−0.142
14 → 15	0.000	−0.061	−0.139
15 → 16	0.000	−0.066	−0.132

d) Dyn, CL

Fig. 1. Experiments

rewarding users (at week 4), the DA values further increase. More in detail, both for DYn and CS analysis global nonconformity is not a significant factor. The values reported for the other two factors are, on the whole, significant, but are uniformly smaller in magnitude for dynamic analysis compared to those of the corresponding weeks in the CS analysis and are less precisely estimated (as represented by uniformly greater standard errors). This is because rather than embodying the structure of the whole network, they embody only the structure of changes in the network over the week, thus the only important value to rely on is the weekly changes of values. This means that "instant" social effects (e.g., friendship reciprocation) have been absorbed into the Week 0 observation, which is not modeled in the dynamic analysis.

Moreover, above mentioned phenomena can be considered as a kind of social "envy": users tend to decrease the scores of other users in order to get more requests for themselves and thus getting more rewards. We note that also LNC value is high while GNC is not impressive: in a sense, users tend to agree on the general topics but not on the specific ones. Similar observations can be made for all the analysis reported in Tables c and d. As a final note, we can observe that the results obtained when sampling data by clustering are slightly better. This behaviour can be explained considering that when leveraging cluster approaches, it is more likely to obtain more homogeneous groups (i.e. group of user sharing common interests and features).

5 Conclusion and Future Work

The impressive amount of data shared on the WEB contains a lot of information regarding user suggestions and searches. In this paper we focus on the effect of users suggestions on their social environment. We showed how personalized suggestions can be extracted by using smart mapping rules. This kind of rules, initially developed for our application scenario, can be profitable used in different contexts. For instance, they allow us to define the fragment of data posting framework, where the data posting problem becomes polynomial. We leveraged many interesting mathematical tools to model several psychological and social mechanisms that proved to be effective in our scenario. We are aware that, with respect to population-scale networks, the networks considered in our experiments are fairly small. However, this aspect does not decrease the validity of our approach as ranking data of the form analyzed here is typically of interest only in specific networks, that are by nature fairly small. Thus, highly scalable techniques are less compelling in our setting. Nevertheless, computationally scalable estimation is an interesting challenge for future research in this area and we want to address it in next future.

References

1. D. Agrawal et al. Challenges and opportunities with big data. A community white paper developed by leading researchers across the United States. 2012.
2. E. Bazzi, N. Cassavia, D. Chiggiato, E. Masciari, D. Saccà, A. Spada, and I. Trubitsyna. Evaluating user behaviour in a cooperative environment. *Information*, 9(12):303, 2018.
3. L. D. Brown. Fundamentals of statistical exponential families with applications in statistical decision theory. *Lecture Notes Monograph Series*, 1986.
4. L. Caroprese and E. Zumpano. Aggregates and priorities in P2P data management systems. In *Proc. of (IDEAS 2011), Lisbon, Portugal*, pages 1–7, 2011.
5. L. Caroprese and E. Zumpano. Computing a deterministic semantics for P2P deductive databases. In *Proc. of IDEAS 2017, Bristol, UK*, pages 184–191, 2017.
6. N. Cassavia, S. Flesca, M. Ianni, E. Masciari, and C. Pulice. Distributed computing by leveraging and rewarding idling user resources from P2P networks. *J. Parallel Distrib. Comput.*, 122:81–94, 2018.
7. N. Cassavia, E. Masciari, C. Pulice, and D. Saccà. Discovering user behavioral features to enhance information search on big data. *TiiS*, 7(2):7:1–7:33, 2017.
8. N. Fiorentino, S. Greco, C. Molinaro, and I. Trubitsyna. ACID: A system for computing approximate certain query answers over incomplete databases. In *Proc. of SIGMOD Conference 2018, Houston, TX, USA*, pages 1685–1688, 2018.
9. S. Greco, C. Molinaro, and I. Trubitsyna. Computing approximate query answers over inconsistent knowledge bases. In *Proc. of IJCAI 2018, Stockholm, Sweden.*, pages 1838–1846, 2018.
10. S. Greco, C. Molinaro, and I. Trubitsyna. Approximation algorithms for querying incomplete databases. *Information Systems (2019)*, https://doi.org/10.1016/j.is.2019.03.010, 2019.
11. P. N. Krivitsky and C. T. Butts. Exponential-family random graph models for rank-order relational data. *http://arxiv.org/abs/1210*.
12. T. Lukasiewicz, E. Malizia, and C. Molinaro. Complexity of approximate query answering under inconsistency in datalog+/-. In *Proc. of IJCAI 2018, Stockholm, Sweden.*, pages 1921–1927, 2018.
13. E. Masciari, D. Saccà, and I. Trubitsyna. Simple user assistance by data posting. In *Second IEEE International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2019, Cagliari, Sardinia, Italy, June 3-5, 2019*, pages 1–8, 2019.
14. E. Masciari, D. Saccà, and I. Trubitsyna. Simplified data posting in practice. In *Proc. of the 23rd International Database Engineering and Applications Symposium (IDEAS 2019), June 10 - 12, 2019, Athens, Greece*, pages 1–6, 2019.