

User group analytics: hypothesis generation and exploratory analysis of user data

Behrooz Omidvar-Tehrani, Sihem Amer-Yahia, Ria Mae Borromeo

▶ **To cite this version:**

Behrooz Omidvar-Tehrani, Sihem Amer-Yahia, Ria Mae Borromeo. User group analytics: hypothesis generation and exploratory analysis of user data. The VLDB Journal, 2019, 28 (2), pp.243-266. 10.1007/s00778-018-0527-4 . hal-02347341

HAL Id: hal-02347341
https://hal.science/hal-02347341

Submitted on 21 Nov 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Author's personal copy

The VLDB Journal
https://doi.org/10.1007/s00778-018-0527-4

REGULAR PAPER

User group analytics: hypothesis generation and exploratory analysis of user data

Behrooz Omidvar Tehrani[1] **Sihem Amer Yahia**[1] **Ria Mae Borromeo**[2]

Received: 6 February 2018 / Revised: 25 September 2018 / Accepted: 13 October 2018
© Springer-Verlag GmbH Germany part of Springer Nature 2018

Abstract
User data is becoming increasingly available in multiple domains ranging from the social Web to retail store receipts. User data is described by user demographics (e.g., age, gender, occupation) and user actions (e.g., rating a movie, publishing a paper, following a medical treatment). The analysis of user data is appealing to scientists who work on population studies, online marketing, recommendations, and large-scale data analytics. User data analytics usually relies on identifying group-level behavior such as "Asian women who publish regularly in databases." Group analytics addresses peculiarities of user data such as noise and sparsity to enable insights. In this paper, we introduce a framework for user group analytics by developing several components which cover the life cycle of user groups. We provide two different analytical environments to support "hypothesis generation" and "exploratory analysis" on user groups. Experiments on datasets with different characteristics show the usability and ef ciency of our group analytics framework.

Keywords User data analytics · User group analytics · Hypothesis generation · Exploratory analysis

1 Introduction

Nowadays, user data is ubiquitous in various domains ranging from the social Web to medical records, scienti c publications, and retail store receipts. This data is the conjunction of *user demographics* (e.g., gender, profession, birth year) and *user actions* (e.g., rating a movie, publishing a paper, following a medical treatment, expressing a political view). Analysis of such data enables novel insights in various scenarios such as population studies [1], online recommendation [2] and targeted advertisement [3].

User data analytics is de ned as a collection of methods and tools to extract value from user data. It relates to a special eld of business analytics, referred to as behavioral analytics [4,5]. The goal of behavioral analytics is to unveil insights into the behavior of consumers on eCommerce platforms, IoT and mobile applications. User data analytics is a need for analysts in their role as *data scientists* who seek to conduct large-scale population studies, and gain insights into various population segments. It is also appealing to users in their role as *information consumers* who use the social Web for routine tasks such as nding a book club or choosing a restaurant. It is also useful to *domain experts* who seek to understand their users and actions.

Most of the current user data analysis approaches target "Quanti ed-Self," i.e., the typical mindset of user data analytics with the aim of exploiting massive personal datasets for self-discovery [6]. In this paper, we shift toward "Quanti ed-Us"[1] and propose user group analytics (UGA), whose aim is to aggregate users into groups to gain a more focused understanding of their behavior. UGA helps analysts make better and faster decisions [7] with more certainty [8].

1.1 Desiderata of UGA

We believe that the following desiderata should be satis ed in UGA in order to obtain meaningful aggregations (i.e., user

Behrooz Omidvar-Tehrani
behrooz.omidvar-tehrani@univ-grenoble-alpes.fr

Sihem Amer-Yahia
sihem.amer-yahia@univ-grenoble-alpes.fr

Ria Mae Borromeo
rhborromeo@up.edu.ph

1 CNRS, Université Grenoble Alpes, Grenoble, France

2 University of the Philippines Open University, Laguna, Philippines

[1] http://www.wired.com/2014/04/forget-the-quanti ed-self-we-need-to-build-the-quanti ed-us/.

Published online: 26 October 2018

Author's personal copy

groups) which enable improved discoveries, pattern identi - cation and meaningful recommendations.

- *Sparsity reduction* Oftentimes, user data is sparse, i.e., many pieces of information for different individual users are missing. Aggregation of user data (i.e., grouping) should contribute to sparsity reduction.
- *Noise reduction* User data may be noisy, i.e., contains wrong information for individual users. User groups should reduce the effect of noise.
- *Improved analysis* Grouping users should unveil new insights which can be employed "internally" by each group member to make better decisions, and "externally" by analysts to make better strategies for the whole group.

We build user aggregations (i.e., groups) based on "frequency." Each generated user group has a certain amount of frequency (above a threshold) among all its members. Hence, a frequent group dissipates negligible values that potentially represent wrong or missing information. The UGA framework employs this simple notion of aggregation in order to reduce noise and sparsity in user data and obtain better insights. However, UGA is not a "data cleaning" contribution and systemic mistakes in user data can be potentially propagated to user groups. In [9], we discuss how an ETL process should precede user group analytics.

1.2 Challenges of UGA

UGA can be very expensive due to the exponential number of possible groups. Any set of users with at least one attribute or action in common can form a group. For instance, we consider a dataset of researchers from 100 different universities who publish in 100 conferences on 100 different topics. In this case, the number of possible groups becomes 2^{300} (i.e., all possible combinations of conferences, topics and universities) which is roughly in order of 10^{40}. This brings the prominent challenge of "information overload": it is a tedious and nearly infeasible task for the analyst to verify all possible groups. Hence, we need to address the two following challenges in UGA.

Semantics By keeping only an interesting subset of groups, the analysis task becomes more manageable (i.e., less information overload). To de ne the "interestingness" of a user group, we propose *quality dimensions* [10], i.e., functions which take a user group as input and return a scalar score.

Navigation Incorporating quality dimensions may not fully address information overload, as there may still exist many interesting groups to verify. Hence there should be a seamless navigation mechanism which enables analysts to walk through the group space and explore one or several groups of interest. Note that while MDL-based summarization approaches tackle information overload as well [11], the lossy compressions and/or high-level abstractions may put burden on analysts for understanding the group space.

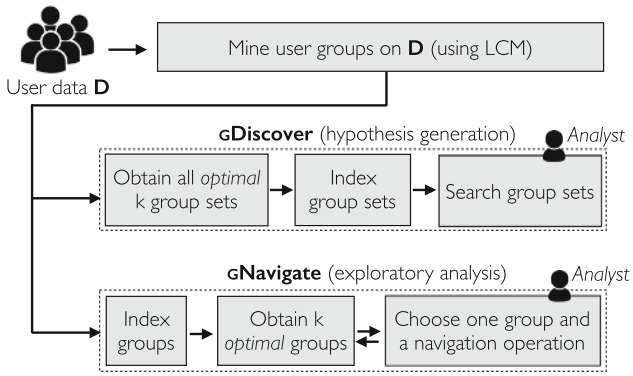


Fig. 1 UGA Framework

1.3 Applications of UGA

In this paper, we describe a UGA framework which targets the aforementioned challenges. We incorporate several analytical components into a unique framework which covers the life cycle of user groups. Figure 1 illustrates our framework. It starts by forming user groups out of raw user data. Once groups are materialized, our framework serves two major applications in UGA, i.e., "hypothesis generation" and "exploratory analysis," described as follows.

Hypothesis generation. In hypothesis generation, the analyst is after collecting evidences in user data which support the hypothesis testing phase. For this aim, our proposed UGA framework generates, stores and indexes all "interesting" sets of user groups. Then a search interface (as the interaction means) is made available for the analyst to look for her interests among generated group sets. Example 1 describes a realistic use case.

Example (Hypothesis generation) It is generally believed that Western-genre movies (e.g., Unforgiven, 1992) are mostly watched by the older generation. This observation is based on demographics breakdown reports on IMDb website.[2] Anna, a social scientist wants to verify this belief by generating possible hypotheses (in form of user groups) for users who watched Western movies. For instance, she obtains the following groups: old male users, young female users, and middle-age users from Texas. By observing those groups, Anna nds that although the overall belief is valid, it also depends on other demographic attributes like gender and location. More precisely, found groups show that not all old people prefer Western movies.

Exploratory analysis It is common that analysts do not have a clear understanding of their needs on user data or it is par-

[2] Internet Movie Database: http://www.imdb.com.

Author's personal copy

tially formulated. In this case, the analyst needs to navigate through various sets of user groups in order to build a knowledge around her interests toward target groups. For this aim, UGA populates interesting group sets on-the-go. At each step of the navigation, the analyst interacts with UGA to reflect her preferences, based on which a set of interesting groups will be generated. Example 2 describes a realistic example.

Example 2 (Exploratory analysis) Tiffany wants to nd a person she met at last night's party in Westford, Massachusetts (MA). She doesn't remember his name or any other indicating contact. Hence no querying mechanism is of help. Tiffany employs UGA to inspect the list of Mike's friends (Mike is the party host.) The UGA framework returns the following groups: engineers in MA who work in NextWorth company and engineers in bioinformatics. This reminds Tiffany that the person was talking about "data visualization," thus he should not be working for NextWorth, a recycling company. Hence she selects the other group. This preference leads her to other relevant groups where she notices a group of software engineers in BioView (a cell imaging company) where she nds the person she was looking for.

1.4 Contributions of UGA

UGA components provide semantic-based and navigation-based solutions to prevent information overload. We implement effective semantics for "hypothesis generation" by enabling a birds-eye-view on all interesting group sets, and effective navigation means for "exploratory analysis" by enabling preference-based interactions with groups. Our UGA framework makes the following contributions.

- We introduce a component for "hypothesis generation" in UGA by formalizing the problem of *discovering interesting group sets*. We introduce following quality dimensions to quantify the interestingness of group sets: coverage, diversity and rating diameter. As all group sets should be materialized before analyst consumption, they should all have optimal values on all quality dimensions. Hence, we formalize the problem as a constrained multi-objective optimization problem with quality dimensions as objectives. We develop an -approximation algorithm (called -DISCOVER) and a heuristic algorithm (called h-DISCOVER) as solutions for hypothesis generation.
- We introduce a component for "exploratory analysis" in UGA by formalizing the problem of *navigating the user group space*. For an effective navigation, we propose GNAVIGATE, an interactive analysis framework on user groups based on simple yet powerful group navigation operations which enable the exploratory analysis of user groups.

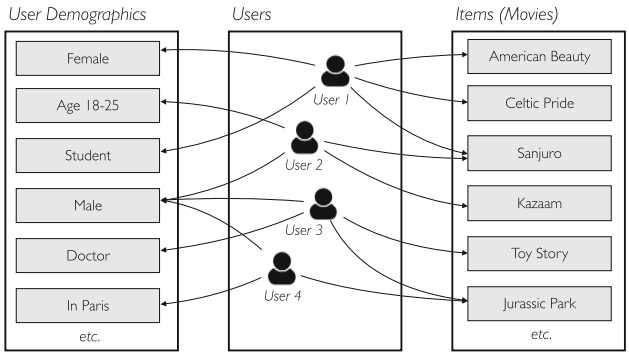


Fig. 2 The gure shows that user 1 is a female student and has watched movies American Beauty, Celtic Pride and Sanjuro. It also shows that the group of male users consists of users 2, 3 and 4, while the group of Jurassic Park watchers contain users 3 and 4

- In an extensive set of experiments, we illustrate the ef - ciency and effectiveness of our UGA framework. In a comprehensive user study, we show the added value of group-based analytics. We also evaluate the performance of our components and provide data-centric insights for real applications of group analytics. All experiments are validated on several user datasets.

The rest of the paper is organized as follows. Section 2 covers preliminary de nitions. Sections 3 and 4 discuss our proposed UGA components (for hypothesis generation and exploratory analysis, respectively). A set of quantitative and qualitative experiments is presented in Sect. 5. We review related work in Sect. 6. Last, we conclude and discuss our future work in Sect. 7.

2 User group model

User data User data contains a set of users $\mathcal{U}$, a set of items $\mathcal{I}$, and a database $\mathcal{D}$ of tuples $\langle u, c, i\rangle$ where $u \in \mathcal{U}$ and $i \in \mathcal{I}$ and c is an action. A tuple $\langle u, c, i\rangle$ represents the action c (such as *authored*, *recorded*, *rated*, *purchased*, *tagged*, *voted*, *followed treatment*) performed by user u on item i. For instance, the tuple $\langle$*John*, *watched*, *Titanic*$\rangle$ means that the user John has watched the movie Titanic. We don't mention actions wherever they are clear from the context, hence a tuple becomes $\langle u, i\rangle$. Our data model can be seen as a bipartite graph centered on users, having demographics on one side and items on the other side (Fig. 2).

User attributes Each user u is described with attributes drawn from a set $_u$ representing demographics information such as "gender" and "age." We also associate a set of attributes to $\mathcal{I}$ denoted as $_i$ representing item details such as "director" for a movie. We refer to the set of all attributes as $= {}_u \cup {}_i$ and each attribute $a_i \in$ has values in $\{v_i^1 \ldots v_i^j \ldots\}$. The domain of values for attribute a_i is

Author's personal copy

denoted as D_{a_i} with $D = \cup D_{a_i}$. For example, if we use a_1 to refer to the gender attribute, it takes two values v_1^1 and v_1^2 representing "male" and "female," respectively.

User datasets Multiple datasets could be represented in our model. Examples are 1.7M research publishing activities of database researchers [3], 5B tweets [12], 300M customer receipts from a retail chain of 1800 stores [13], 10M rating records from MovieLens [14], 50M artist ratings from LastFM [15], 1M electronic health records (EHR) [16] and 200K book ratings from BookCrossing [17]. We provide more details about the datasets that we use in this paper in Sect. 5.1.

Now, we formally define user groups as follows.

Definition 1 (*User Group*) A group g is a set of tuples $\langle u, i \rangle \in \mathcal{D}$ where $u \in \mathcal{U}' \subseteq \mathcal{U}$ and $i \in \mathcal{I}' \subseteq \mathcal{I}$ identified with a label $g = [P_g^u, P_g^i, \mathcal{I}']$ where P_g^u and P_g^i are conjunctions of predicates on user attributes and item attributes, respectively. Each user in $\mathcal{U}'$ must satisfy P_g^u and $\forall i \in \mathcal{I}', i$ satisfies P_g^i.

Based on Definition 1, a user group $g_1 = [\langle$gender, female$\rangle$, $\langle$movie, Titanic$\rangle]$ contains females who watch (or rate) the Titanic movie. When attributes are clear from the context, we usually show group labels in a more concise form, e.g., $g_1 =$ [female, Titanic]. Group labels express the behavior and common demographics of group members. For a group g, we use the notion $|g|$ to denote the number of members in that group.

We use $\mathcal{G}$ to refer to the set of all user groups forming a group space. In UGA, we assume $\mathcal{G}$ is already pre-computed using any group generation algorithm. $\mathcal{G}$ is often very large, as it is exponential in the number of items and attribute values (i.e., the information overload problem). We employ LCM pattern mining algorithm for generating groups [18]. Given a frequency threshold σ, each mined frequent pattern of LCM corresponds to a user group with at least σ users. To feed LCM, we convert attribute value pairs in group labels into an item. For instance, $\langle$gender, male$\rangle$ and $\langle$gender, female$\rangle$ become two independent items. Note that UGA is independent from the group generation process and can virtually leverage any other method, such as clustering, community detection and team formation.

3 Hypothesis generation on user groups

Our first application of UGA is hypothesis generation. While statistical validations using hypothesis testing are employed to yield a p value representing the extremity of observations under a null hypothesis (e.g., detecting false discoveries [19] and Simpson paradoxes [20]), hypothesis generation seeks clues to elaborate the null hypothesis and enable evidence-based decision making using user groups. Hypothesis generation relies on the problem of discovering interesting user group sets in UGA framework, referred to as GDiscover. In practice, there does not exist analytical tools that enable the scalable, on-demand discovery of user groups. Our aim is to generate *all interesting group sets* of users in order to facilitate hypothesis generation on user data.

3.1 Desiderata of GDiscover

We define desiderata that user groups should satisfy (local desiderata) and those that must be satisfied by the group set (global desiderata). Local desiderata are as follows:

- *Describability* Each group should be easily understandable by the analyst. While this is often difficult to satisfy through unsupervised clustering of users, it is easily enforced in our approach since each group must conform to Definition 1 (definition of user groups) and provide a label.
- *Size* Returning groups that contain too few members is not meaningful to the analyst. We hence need to impose a minimum frequency constraint on groups.

Global desiderata are as follows:

- *Coverage* Together, returned groups should cover most users in the user data. While ideally we would like each and every user to belong to at least one group, that is not always feasible due to other local and global desiderata associated with the set of returned groups.
- *Diversity* Returned groups need to be different from each other in order to provide complementary information on users.
- *Distribution* The behavior of members in selected groups should follow a requested distribution (e.g., being homogeneous or being polarized).
- *Number of groups* The number of returned groups in group sets should not be too high in order to provide the analyst with an at-a-glance understanding of the user data.

The following example demonstrates how the desiderata defined above enable hypothesis generation.

Example 3 (Online advertising) We consider the process of finding the best target audience for an online advertisement. Amber, an advertising agent, is tasked to find the best target audience worldwide for a promotion on the new edition of the book "Rumi's Secret" by Brad Gooch. Based on previous best-selling reports,[3] she hypothesizes that the best place to

[3] Jane Ciabattari: Why is Rumi the best-selling poet in the US? http://www.bbc.com/culture/story/20140414-americas-best-selling-poet.

Author's personal copy

look for this target audience is inside the USA. To nd a target group, Amber goes to BOOKCROSSING website[4] and focuses on the set of 3800 reviewers of the book. Amber observes that 89% of those users are **covered** with a group set whose groups are "young reviewers in New York City," "middle-age reviewers in Seattle," and "old females in Denver." Beyond coverage, these three groups are also **diverse**, i.e., they do not overlap because their reviewers belong to different age categories. Amber nds the second group promising, as it is a homogeneous group (i.e., most users vote their reviews with a high score). Thus, it is a candidate group for audience targeting.

3.2 Challenges of GDISCOVER and solution overview

The principled challenge for GDISCOVER is to quickly identify a group set that satis es local and global desiderata. That is a hard problem because of two following reasons:

- *Huge candidate set* First the pool of candidate group sets is very large. Any possible combination of attribute value pairs and items can form a group, and any number of groups can form a group set.
- *Need for multi-objective optimization* The second reason of hardness is that our user-centered objectives (i.e., coverage, diversity and distribution) are conflicting objectives. That means optimizing one does not necessarily lead the best values for others. Thus, the need for a multi-objective optimization approach that will not compromise one objective over another. Such an approach would return *the set of all candidate group sets* that are not dominated by any other along all objectives.

To tackle aforementioned challenges, we propose -DISCOVER, an -approximation algorithm which satis es local and global desiderata and ensures to nd group sets that are -far from optimal ones. Since -DISCOVER relies on an exhaustive search in the space of all groups, we also propose *h*-DISCOVER, a heuristic that exploits the lattice formed by user groups and makes maximal pruning in order to speed up the GDISCOVER process.

3.3 Group quality dimensions

We now formalize our user-centered objectives. We extend a tuple $u, i\rangle$ with attributes of u and i (i.e., $_u$ and $_i$, respectively) and a numerical rating score s, hence the tuple becomes $a_1, a_2, a_3 \ldots, s\rangle$.[5] We are given a subset of users $U \subseteq \mathcal{U}$ and a group set $G \subseteq \mathcal{G}$.

Coverage is a value between 0 and 1 and measures the proportion of users in U who are present in groups in G. Coverage guarantees the quality of completeness, i.e., how much of the input users (i.e., U) match with G.

$$coverage(G, U) = |\cup_{g \in G} (u \in U, u \in g)|/|U| \quad (1)$$

For instance, in Fig. 3, $coverage(G, U) = 0.8$ where $G = \{g_1, g_2\}$ and U describes Toy Story watchers.

Diversity is a value between 0 and 1 that measures how distinct groups in group set G are from each other. Diversity penalizes (exponentially) group sets containing overlapping groups.

$$diversity(G, U) = 1/(1 + \Sigma_{g,g' \in G} | u \in U, u \in g \wedge u \in g'|) \quad (2)$$

For instance, in Fig. 3, $diversity(G, U) = 0.5$. By convention, if $|G| = 1$, we consider $g' = U$ in Eq. 2 which leads the lowest possible diversity value. Diversity is useful to obtain different aspects of the input subset of users. Note that among different ways of de ning diversity (e.g., Jaccard, Cosine and Shannon Entropy), we chose the one in Eq. 2 to penalize overlaps exponentially.

Rating diameter A group set G may be characterized by its rating distribution. The rating distribution of a single group g is a histogram with cardinalities of rating scores provided by g's members. There are different ways of quantifying a rating distribution. In this work, we consider a simple-to-interpret linear measure called "rating diameter" which computes the difference between the highest and the lowest rating scores in a group (Eq. 3).

$$diameter(G) = average_{g \in G}(max_{u \in g}(u.s) - min_{u' \in g}(u'.s)) \quad (3)$$

In Fig. 3, $diameter(G) = 3$. The rating diameter in a group provides analysts with the ability to tune the quality of found groups according to speci c needs. Example 3 is a good case for homogeneity: by reporting a low diameter for the group of middle-age reviewers in Seattle, Amber understands that most individuals in that group have monotonous views on the Gooch's book. Generally, a user group may exhibit different rating distributions:

- *Homogeneous* A homogeneous rating distribution shows that all users in G have approximately agreed on a unique

[4] http://www.bookcrossing.com.

[5] Note that this is just the way we illustrate the concept and of course we do not make this concatenation on the user data in the database.

Author's personal copy

ID	Movie	Name	Gender	Age	Occup.	Rating
u_1	Toy Story	John	M	young	teacher	4
u_2	Toy Story	Jennifer	F	old	teacher	3
u_3	Toy Story	Mary	F	old	teacher	2
u_4	Titanic	Carine	F	old	other	4
u_5	Toy Story	Sara	F	young	student	3
u_6	Toy Story	Martin	M	young	student	5
u_7	Titanic	Peter	M	young	student	1

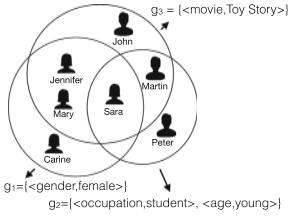


Fig. 3 Illustration of group sets. The gure illustrates 7 users and 3 groups. For instance, g_1 represents 4 female reviewers, and g_2 contains 3 young students. Note that there exists one member in common between the two mentioned user groups

score. We use this rating distribution when we are seeking a consensus between group members and to provide a *representative unique score* for the whole group set. An example for this rating distribution is the movie The Godfather in IMDb, as 53.7% of ratings are for the highest score.[6]

- *Balanced* A balanced rating distribution shows that the preference of group members is equally distributed among scores. A user group with balanced rating distribution counts as a "neutral group": there is no preference for any score. A neutral group can be used as a reference to see how other groups are biased toward a score.
- *Polarized* A polarized rating distribution shows that group members have the farthest possible preferences from each other. A real example for this rating distribution is the movie Fifty Shades of Grey in IMDb, as 28.8% and 15.9% of ratings are for the lowest and highest scores, respectively.[7]

Based on De nition 3, a small value of $diameter(G)$ leads a homogeneous group set G and a high value leads a polarized group set G.

3.4 Multi objective optimization principles

We propose to use the quality dimensions de ned in Sect. 3.3 as optimization objectives. When dealing with more than one dimension to optimize, there may be many incomparable group sets. For instance, for a subset of users $U \subseteq \mathcal{U}$, we can form two group sets, G_1 with $coverage(G_1, U) = 0.8$ and $diversity(G_1, U) = 0.4$ and G_2 with $coverage(G_2, U) = 0.5$ and $diversity(G_2, U) = 0.7$. Each group set has its own advantage: the former has higher coverage and the latter has higher diversity. Another group set G_3 with $coverage(G_3, U) = 0.5$ and $diversity(G_3, U) = 0.2$ has no advantage compared to G_1, hence it can be ignored. In other words, G_3 is dominated by G_1. In this section, we borrow the terminology of multi-objective optimization and de ne these concepts more formally.

Definition 2 (*Plan*) A plan p_i, associated to a group set G_i for a subset of users U, is a tuple $\langle |G_i|, coverage(G_i, U), diversity(G_i, U), diameter(G_i) \rangle$.

Definition 3 (*Sub-plan*) A plan p_i is the sub-plan of another plan p_j if their associated group sets satisfy $G_i \subseteq G_j$.

Definition 4 (*Dominance*) Plan p_1 dominates p_2 if p_1 has *better* or equivalent values than p_2 in every objective. The term *"better"* is equivalent to *"larger"* for *maximization* objectives (e.g., diversity, coverage and polarization), and *"lower"* for *minimization* ones (e.g., homogeneity). Furthermore, plan p_1 strictly dominates p_2 if p_1 dominates p_2 and the values of objectives for p_1 and p_2 are not equal.

Definition 5 (*Pareto Plan*) Plan p is Pareto if no other plan strictly dominates p.

In the example above, plan p_2 that corresponds to G_2 dominates p_3 (for G_3) and plan p_1 (for G_1) strictly dominates p_3. Furthermore, p_1 and p_2 are Pareto plans. The set of all Pareto plans is denoted as $\mathcal{P}$.

3.5 GDISCOVER problem definition

We de ne the GDISCOVER problem as a constrained multi-objective optimization problem: for a given subset of users U and integer constants σ (frequency threshold) and k (size threshold), the problem is to identify all group sets, such that each group set G satis es:

- $coverage(G, U)$ is maximized;
- $diversity(G, U)$ is maximized;
- $diameter(G)$ is optimized;
- $|G| \leq k$;
- $\forall g \in G : |g| \geq \sigma$.

The last constraint states that a group g should contain at least σ users, an application-de ned frequency threshold. Note that while we always maximize coverage and diversity, we may either minimize (e.g., in case of homogeneity) or maximize (e.g., in case of polarization) the diameter objective based on the analyst's needs. We stress the fact that we optimize only one diameter objective at a time. We state the complexity of our problem in "Appendix."

3.6 GDISCOVER algorithms

The main challenge in designing an algorithm for our problem is the multi-objective nature of the problem. A multi-objective problem can be easily solved in two following cases.

[6] http://www.imdb.com/title/tt0068646/ratings?ref_=tt_ov_rt.

[7] http://www.imdb.com/title/tt2322441/ratings?ref_=tt_ov_rt.

Author's personal copy

Algorithm 1: -DISCOVER algorithm

```
   Input: k,    > 1, U
1  P   ← ∅;
2  for all user groups g (already materialized for U) do
3  |  p_g ← construct_plan(g);
4  |  if p_g is not  -dominated by any other plan in P  then
   |  P .add(p_g)
5  end
6  for n ∈ [2, k] do
7  |  for group sets G of size n  do
8  |  |  p_G ← construct_plan(G);
9  |  |  if p_G is not  -dominated by other plans in P  then
   |  |  P .add(p_G)
10 |  end
11 end
12 return P
```

Algorithm 2: h-DISCOVER algorithm

```
   Input: σ, k,  , U, n (number of iterations)
1  P_h ← ∅
2  N ← Set of intervals on diversity values
3  for n times  do
4  |  G_s ← random_groupsets(k, σ, U)
5  |  G*_s ← SHC(G_s, U)
6  |  interval ← get_interval(G*_s)
7  |  N[interval].add(G*_s)
8  end
9  for interval ∈ N  do
10 |  Keep non-dominated plans in interval and add them to P_h
11 end
12 P_h ← optimize_diameter(P_h)
13 return P_h
```

– *Scalarization* if it is possible to combine all objective dimensions into a single dimension and use typical single-objective optimization algorithms (e.g., Randomized Hill Climbing Exploration);
– *Consistent objectives* if optimizing one dimension leads an optimized value for other dimensions.

First, it is not possible in GDISCOVER problem to combine all objective dimensions into a single dimension [21]. We provide an intuition of the reason as follows. Let us consider the *sum* aggregation function to combine coverage and diversity values of a plan into a single score. Let p_1 and p_2 be two plans corresponding to two group sets G_1 and G_2, respectively, and $coverage(G_1, U) = 0.5$, $diversity(G_1, U) = 0.8$, $coverage(G_2, U) = 0.6$ and $diversity(G_2, U) = 0.1$. In this case, the score of p_1 is 1.3 and the score of p_2 is 0.7. Hence, we would prune p_2, while it has a higher value for coverage.

Second, our objectives are *conflicting*, i.e., optimizing one does not necessarily lead to optimizing others. We denote a group set that optimizes all quality dimensions at a same time, as *zenith group set*. Achieving the zenith group set is infeasible in almost all problems. For instance, a group set containing $g_1 =$ [female, young] and $g_2 =$ [student, young] has a high coverage for the American Beauty reviewers. However, its diversity is not high, because many young users are also students.

In this paper, we discuss 3 different algorithms for our problem: exhaustive, approximation and heuristic.

3.6.1 Exhaustive and approximation algorithms

The exhaustive algorithm starts by calculating Pareto plans for single groups. Then it iteratively calculates plans for group sets containing more than one group by combining single groups. At each iteration, dominated plans are discarded. The algorithm combines sub-plans to obtain new plans and exploits the *optimality principle* (POO) for pruning [22]. This approach makes an exhaustive search over all combinations of groups to nd Pareto plans, i.e., both time- and space-consuming [21].

We propose to improve the complexity of the exhaustive algorithm with our *approximation-based* algorithm which makes less enumerations and guarantees the quality of results. Another way of improvement is *heuristic-based* which will be discussed in Sect. 3.6.2. For our approximation algorithm, we exploit the near-optimality principle (PONO) [22]. For simplicity, we use $f(G)$ to denote the value of an objective function f for a group set G.

Definition 6 (*PONO*) Given an objective f and ≥ 1 (is a precision value), derive G' from G by replacing G_1 by G'_1 and G_2 by G'_2. Then $f(G'_1) \geq f(G_1) \times$ and $f(G'_2) \geq f(G_2) \times$ together imply $f(G') \geq f(G) \times$.

We formally prove that all our objectives (coverage, diversity and rating diameter) satisfy PONO [23]. As PONO overrides POO, a new notion of dominance is introduced in De nition 7 to be in line with PONO.

Definition 7 (*Approximated Dominance*) Let ≥ 1, a plan p_1 -dominates p_2 if for every objective f, $f(G_1) \geq f(G_2) \times$ where $f \in \{diversity, coverage, polarization\}$ and $f(G_1) \leq f(G_2) \times$ where f is *homogeneity*.

Definition 8 (*Approximated Pareto Plan*) For a precision value , plan p is an -approximated Pareto plan if no other plan -dominates p.

It is shown in [22] that *generating less plans makes a multi-objective optimization algorithm run faster*. This is because the execution time heavily depends on the number of generated plans. Thus, a pruning strategy dictated by PONO is at the core of an approximation algorithm for multi-objective optimization.

We adapt the -approximation algorithm proposed in [22] to the context of our problem and propose -DISCOVER (Algorithm 1). The main idea is to exploit a bottom-up

Author's personal copy

Algorithm 3: Shotgun hill climbing (SHC)

```
   Input: Group set G, U
1  G* ← ∅
2  while true do
3  |  C ← ∅
4  |  for g ∈ G and each lattice-based parent g' of g do
5  |  |  G' ← G − {g} + {g}'
6  |  |  C.add(G', coverage(G', U))
7  |  end
8  |  let (G'_m, coverage(G'_m, U)) be the pair with maximum
   |  coverage
9  |  if coverage(G'_m, U) ≤ coverage(G, U) then
10 |  |  G* ← G
11 |  |  return G*
12 |  end
13 |  G ← G'_m
14 end
```

dynamic programming approach. The algorithm begins by constructing a plan for each single-user group (lines 2–5). We keep all non- -dominated plans of single groups in a buffer. Then it builds group sets of size 2 up to size k using plans in the buffer (lines 7–11). After each iteration, we remove -dominated plans from the buffer. At the end, we return the buffer content. This approach creates a tree between group sets, which we simply call *group set tree*. For instance in a group set tree, two group sets $\{g_1, g_2\}$ and $\{g_3, g_4\}$ are children of the parent group set $\{g_1, g_2, g_3, g_4\}$.

The crucial part of Algorithm 1 is its pruning mechanism using the precision value . In the special case of $= 1$, the algorithm operates exhaustively. If > 1, the algorithm prunes more and hence is faster. In the latter case, a new plan is only compared with all plans that generate the same result. But a new plan is only inserted into the buffer if no other plan approximately dominates it. This means that -DISCOVER tends to insert fewer plans than the exhaustive algorithm. The exhaustive algorithm inserts new plans if they do not fall within the dominated area, but -DISCOVER inserts new plans if they fall neither into the dominated nor into the approximately dominated area.

3.6.2 Heuristic algorithm

A heuristic algorithm has obviously its own advantages and disadvantages. Although a heuristic algorithm does not provide any approximation guarantee, it eventually returns a subset of Pareto set. Nevertheless, the fact that it generates a subset of the Pareto makes it faster.

Algorithm 2 illustrates our heuristic algorithm h-DISCOVER. The algorithm starts by making n different iterations on nding optimal points to avoid local optima (lines 3–8). Note that n is a user-de ned parameter which de nes the number of efforts to reach the global optima.

At each iteration, the algorithm begins with a random group set of size k called G_s (line 4). Then a Shotgun Hill Climbing local search approach [24] (SHC) is executed (Algorithm 3) to nd the group set with optimal value starting from G_s (line 5). SHC maximizes coverage. Diversity is already divided into intervals $\mathcal{N}$ for each of which a buffer is associated. $\mathcal{N}$ is also user-de ned and relates to the number of results in the output. The resulting group set of SHC is placed in the buffer whose interval matches the diversity value of the group set (line 7). SHC operates on a generalization/specialization lattice whose navigation in a downward fashion satis es a monotonicity property for coverage described in the following theorem.

Theorem 1 *Given any two groups g and g' where g is the parent of g', the coverage of g is no smaller than the coverage of g'.*

Proof Given any two user groups g and g' where g is the parent of g', let l_g denotes g's label. For g' to be the child of g, its label should necessarily contain at least one more attribute–value pair. Thus, $l_{g'}$ should be $l_g \cup \{\langle a_i, v_i^j \rangle\}$, where $\langle a_i, v_i^j \rangle$ is the attribute–value pair which holds for all users in g' but not g. Thus, g covers all users which are covered by g' plus users U for whom $\langle a_i, v_i^j \rangle$ does not hold. Thus, g covers as many users as g' covers or more. □

Finally, n different plans are distributed in different interval buffers. The algorithm then iterates over interval buffers to prune dominated plans (lines 9–11). Based on De nition 4, a plan is pruned and removed from its buffer if it is dominated by other plans. Finally, for each interval, we report one unique plan that has the best value for the requested diameter objective (line 12). The best value for homogeneity is the lowest, while for polarization, it is the highest. If the rating diameter is not speci ed or the rating score does not exist in the user dataset, all plans in the buffer will be returned.

3.7 Group set retrieval

Hypothesis generation in GDISCOVER is performed via a search platform. Once interesting group sets are fully materialized (by using either -DISCOVER or h-DISCOVER), they should be easily and ef ciently accessible for search. Given a set of search terms $\mathcal{S}$, a naïve approach is to scan each group label in each group set and return the most relevant ones. However, this brute-force approach is expensive: all group sets and their internal groups should be compared with the search terms.

To overcome this challenge, we build a *bit signature* $\overrightarrow{G}$ for each group set G which represents the presence (with 1s) and absence (with 0s) of attributes and items in the group label. For a group set G, $\overrightarrow{G}$[Titanic]= 1 iff the item Titanic exists in at least one of labels of G's groups. We build a hash index

Author's personal copy

Algorithm 4: Group set retrieval algorithm

```
Input: Search terms S, hash structure H
min_distance ← ∞
S⃗ ← bit_signatuare(S)
G⃗ ← H[S].next()
while H's end is not met do
    distance ← hamming_distance(G⃗, S⃗)
    if distance < min_distance then
        min_distance ← distance
        G_out ← G⃗
    end
    G⃗ ← H[S].next()
end
return G_out
```

on bit signatures and sort group sets based on their signature values in chronological order. We denote this hash structure as $\mathcal{H}$. Algorithm 4 operates on $\mathcal{H}$ for fast veri cation of group sets against search terms.

Algorithm 4 starts by forming a bit signature for search terms $\mathcal{S}$, to be comparable with group sets. Then the algorithm iterates on group sets in $\mathcal{H}[\mathcal{S}]$ and compares their bit signature with $\mathcal{S}$'s using "hamming distance" function. This is a string metric function for measuring the edit distance of 0s and 1s to transform $\overrightarrow{G}$ into $\mathcal{S}$ [25]. All comparisons are done using bitwise operations, hence is extremely fast.

4 Exploratory analysis on user groups

Unlike hypothesis generation (i.e., GDISCOVER in Sect. 3) and most common clustering methods [26], group sets in exploratory analysis are not materialized in advance, but will be generated on-the-fly based on the analyst's preferences. In the former context, the analyst has an explicit hypothesis in mind to describe in form of search terms. In the latter, the analyst has a partial understanding of the task. To address that, we introduce GNAVIGATE, an interactive user group analysis approach which enables human-in-the-loop inspection of user groups. The framework allows analysts to incrementally discover interesting subsets of user data by exploring relevant groups until landing on target groups. More speci cally, analysts are able to employ GNAVIGATE in two different scenarios: *single-user target*, i.e., nding a speci c user in a group (as in Example 2 where Tiffany is looking for a speci c person) and *multi-user target*, i.e., explore group sets and gather several users who may be scattered in different groups of interest. Those scenarios cannot be performed with GDISCOVER, as there is no way to declaratively de ne the analyst's request which is iterative in nature. GNAVIGATE is built upon three following key principles:

- *P1: The analyst must be able to navigate different groups but not be overwhelmed with many options.* We break the navigation process of GNAVIGATE into successive steps during which an analyst chooses a seed group, examines the users it contains, manipulates group members (by adding/removing users), and continues with the navigation process. This principle is in line with "enumeration" and "insights" principles discussed in [27] for guided interaction.
- *P2: Groups offered to the analyst must be of high quality.* Group sets in GNAVIGATE are not optimized in advance against interestingness measures. Hence, group sets offered at each step should be optimized on-the-fly. In step i of the navigation process, the group set should be *relevant* to the analyst's choice in step $i-1$, and also be as *diverse* and *covered* as possible.
- *P3: The train of thought of the analyst must not be lost.* Each interactive group navigation step must be fast. This is in line with "responsiveness" principle discussed in [27].

4.1 Group navigation primitives

Our navigation primitives, i.e., *explore*() and *exploit*() constitute building blocks for navigating in user groups. We rst de ne *explore*() that is designed to navigate in the group space in an outward way: starting from a set of users, it discovers groups containing new users. Given a subset of users $U \subseteq \mathcal{U}$ and a relevance threshold μ, $explore(U, \mathcal{G}, \mu)$ nds all groups in $\mathcal{G}$ whose overlap with U are at least μ (Eq. 4).

$$explore(U, \mathcal{G}, \mu) = \{(g, overlap(U, g)) | g \in \mathcal{G} \wedge g \neq U \wedge overlap(U, g) \geq \mu\} \quad (4)$$

In Eq. 4, $overlap(U, g) = \frac{|U \cap g|}{|U \cup g|}$ (i.e., Jaccard similarity coef cient). The overlap condition provides a progressive exploration of the space, which helps the analyst build an incremental understanding of the underlying data.

When an interesting group is found, another operation is *exploit*(), i.e., delving into the most interesting subgroups contained in an input group (akin to "drill-down" operator in OLAP for cubes [28]). Given a subset of users $U \subseteq \mathcal{U}$, $exploit(U, \mathcal{G})$ nds all groups in $\mathcal{G}$ that are contained in U (Eq. 5).

$$exploit(U, \mathcal{G}) = \{g \in \mathcal{G} | g \subseteq U\} \quad (5)$$

4.2 GNAVIGATE problem definition

The navigation of user groups relies on the two primitives, *explore*() and *exploit*() (Eqs. 4 and 5, respectively), that are applied to an input group. In order to comply with principles

Author's personal copy

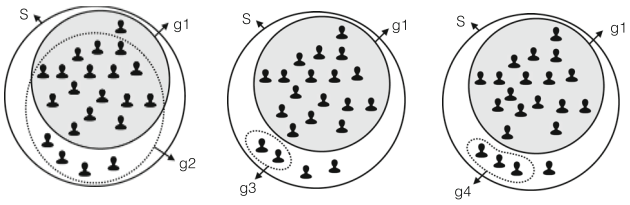


Fig. 4 Incorporating coverage into diversity

P1 and *P2*, the number of groups returned to the analyst at each step must be limited, and output groups must exhibit diversity. Hence, we de ne the GNAVIGATE problem as follows: given a subset of users $U \subseteq \mathcal{U}$, a relevance threshold μ, return k groups in $\mathcal{G}$, referred to as $\mathcal{G}_U$ and is expressed either as an exploration (i.e., opExplore) or an exploitation (i.e., opExploit) problem depending on analyst's needs.

For exploration, we de ne opExplore$(U, \mathcal{G}, \mu, k)$ that must satisfy the following conditions:

- $\mathcal{G}_U \subseteq explore(U, \mathcal{G}, \mu)$
- $|\mathcal{G}_U| = k$
- $diversity(\mathcal{G}_U)$ is maximized, where $diversity(\mathcal{G}_U)$ is de ned in Eq. 6.

$$diversity(\mathcal{G}_U) = \Sigma_{\{g_1,g_2\}\subseteq\mathcal{G}_U|g_1\neq g_2}(1 - overlap(g_1, g_2)). \quad (6)$$

For exploitation, we de ne opExploit$(U, \mathcal{G}, k)$ that must satisfy the following conditions:

- $\mathcal{G}_U \subseteq exploit(U, \mathcal{G})$
- $|\mathcal{G}_U| = k$
- $divCoverage(\mathcal{G}_U)$ is maximized, where $divCoverage(\mathcal{G}_U)$ is de ned in Eq. 7.

$$divCoverage(\mathcal{G}_U) = diversity(\mathcal{G}_U) \times \left(|\bigcup_{g\in\mathcal{G}_U} g|/|U|\right) \quad (7)$$

In opExploit, the aim is to nd k groups that maximize coverage of the seed set U. Choosing k groups that have the highest coverage may potentially cause high overlap between those groups. Figure 4, left, illustrates that, with $k = 2$ and two highly overlapping groups g_1 and g_2. Therefore, in opExploit's case, we revisit the de nition of diversity in a way that it prioritizes k diverse groups which *cover as many users as possible in* U. As there does not exist a unique optimal solution for both diversity and coverage (see Sect. 3 and also [29]), the diversity formula is modi ed by adding $(|\bigcup_{g\in G_U} g|/|U|)$ (see Eq. 7). For example, in Fig. 4, $diversity(\{g_1, g_3\}) = diversity(\{g_1, g_4\}) = 1.0$. Thus, for opExplore, both g_3 and g_4 can be chosen with g_1. However, for opExploit, g_4 is preferred because $divCoverage(\{g_1, g_4\}) > divCoverage(\{g_1, g_3\})$.

In addition to opExplore and opExploit, the analyst is provided with a set of actions that could be performed on a chosen group to transform it according to her needs. The analyst examines the set of k groups at each step and chooses a new input group on which one of the following actions could be performed: keep(U, U'), add(U, l), remove(U, l) and actUndo() to undo the previous step. Table 1 describes each action.

Table 1 GNAVIGATE actions

Action	Description
keep(U, U')	Keeps users U' in U
add(U, l)	Augments l_U with l
remove(U, l)	Removes l from l_U
actUndo()	Back-tracks to their previous step

Algorithm 5: GNAVIGATE algorithm

```
Input: g ∈ 𝒢, op, μ, k, timelimit
1  𝒢_g ← top_k(ℒ^g)
2  g_out ← get_next(ℒ^g)
3  while (timelimit not exceeded ∧ overlap(g, g_out) ≥ μ) do
4      for g_in ∈ 𝒢_g do
5          if better_diversity(𝒢_g, g_out, g_in, op) then
6              𝒢_g ← replace(𝒢_g, g_out, g_in)
7              break
8          end
9      end
10     g_out ← get_next(ℒ^g)
11 end
12 return 𝒢_g
```

4.3 GNAVIGATE algorithm

GNAVIGATE requires an ef cient algorithm for dynamically nding and comparing user groups. In "Appendix," we show that our problem is NP-complete by reductions from the MAXIMUM EDGE SUBGRAPH problem for opExplore and from the MAXIMUM COVERAGE problem for opExploit.

Prior to GNAVIGATE, we pre-compute an inverted index for each user group $g \in \mathcal{G}$ (as is commonly done in Web search) in an offline step in order to speedup computing group relevance. Each index $\mathcal{L}^g$ stores all other groups in $\mathcal{G}$ in decreasing order of their overlap with g. Thanks to the relevance threshold μ, we only partially materialize the indices.

In order to comply with principle *P3*, GNAVIGATE introduces a time limit parameter, i.e., each step of GNAVIGATE solves the group navigation problem (introduced in Sect. 4.2)

Author's personal copy

and returns the best possible k groups within a given time limit.

Algorithm 5 summarizes a single greedy procedure for GNAVIGATE, be it opExplore or opExploit. The algorithm is called at each step of GNAVIGATE. The algorithm admits as input a user group g, an operation op (opExplore or opExploit), a relevance threshold μ, a size threshold k, and a time limit *timelimit*, and returns the best k groups denoted $\mathcal{G}_g$. Line 1 selects the most overlapping groups with g by simply retrieving the k highest ranking groups in $\mathcal{L}^g$ in $\mathcal{O}(1)$. Function $get_next(\mathcal{L}^g)$ (line 2) returns the next group g_{in} in $\mathcal{L}^g$ in sequential order. Lines 3–11 iterate over the inverted indices to determine if other groups should be considered to increase diversity while staying within the time limit and not violating the overlap threshold with the selected group. Since groups in $\mathcal{L}^g$ are sorted on decreasing overlap with g, the algorithm can safely stop as soon as the overlap condition is violated (or if the time limit is exceeded.)

The algorithm then looks for a candidate group $g_{out} \in \mathcal{G}_g$ to replace in order to increase diversity. The boolean function $better_diversity()$ (line 5) checks if by replacing g_{out} by g_{in} in $\mathcal{G}_U$, the overall diversity of the new $\mathcal{G}_U$ increases. Obviously, the diversity of a group set $\mathcal{G}_k$ depends on the operation op.

The number of diversity improvement loops (lines 3–11) is $|\mathcal{L}^g|$ in the worst case. For each group $g_{in} \in \mathcal{G}_g$, we verify if the diversity score is improved by $better_diversity()$, hence $\mathcal{O}(k^2)$. The time complexity of the algorithm is then $\mathcal{O}(k^2.max_{g \in \mathcal{G}}|\mathcal{L}^g|)$.

5 Experiments

To evaluate the ef ciency and effectiveness of our UGA framework, we made an extensive set of quantitative and qualitative experiments. All experiments are performed using our prototype built on JDK 1.8.0 and conducted on an 2.4 GHz Intel Core i5 with 8 GB of memory on OS X 10.13.2 operating system. First, we introduce the datasets used in our experiments (Sect. 5.1). Then we discuss our qualitative experiment in form of a user study for measuring the usefulness of our interestingness measures (Sect. 5.2). Last, we discuss quantitative experiments in Sect. 5.3.

5.1 Datasets

The data model that we introduced in Sect. 2 can be used to model many different user datasets. Table 2 provides a summary of statistics for the datasets used in our experiments. MOVIELENS dataset contains records $\langle u, i \rangle$ representing that user u votes for movie i. A score s is associated to each voting action (based on a 5-star Likert scale). The dataset contains 1,000,209 anonymous votes of 6040 users on 3952 movies. MOVIELENS provides four user attributes: gender, age, occupation and zipcode. We convert the numeric age into four categorical attribute values, namely "teenager" (under 18), "young" (18–35), "middle age" (35–55) and "old" (over 55). We also convert zip-codes to states in the USA (or "foreign," if not in the USA) by using the USPS zip code lookup.[8] This generates the location attribute which takes 52 distinct values.

Table 2 Statistics on datasets

	# users	# items	# attribs.
MOVIELENS	6040	3900	8
DM AUTHORS	4907	11,890	4

DM AUTHORS contains 4907 researchers who have at least 3 publications in one of the following top data management conferences[9]: WWW, KDD, SIGMOD, CIKM, ICWSM, EDBT, ICDM, ICDE, RecSys, SIGIR and VLDB. We crawled authors in October 2014 from DBLP[10] for the period of 2000–2014. A record $\langle u, i \rangle$ in this dataset means that researcher u has contributed to item i where i can be a conference, journal or a keyword (e.g., "data integration"). The main advantage of this dataset is that each user is *known* and *veri able* (unlike anonymous voters in other datasets). Thus, we can check if resulted groups on this dataset are meaningful. For each researcher, we collect following attributes: seniority, number of publications, publication rate, venues, topics and gender. More details on attributes are provided in [30]. The number of all attribute values in DM AUTHORS dataset reaches 11,890. We provide public access to DM AUTHORS dataset.[11]

5.2 Qualitative experiments

Most principled questions regarding the UGA framework are as follows: *"does group-based analytics provide more insights than individualistic analytics?"*, *"if groups are preferred to individual users, are diversity and coverage 'good' measures to capture high quality groups?"* The usefulness of our UGA framework depends on these questions. To nd an answer, we performed a user study in Amazon Mechanical Turk[12] (AMT) with 50 participants (i.e., workers in AMT) and asked them to perform a few tasks on UGA. Some statistics about these participants are mentioned in Table 3. We rely on the state-of-the-art in Information Retrieval and Databases where an agreement was established on the way user studies

[8] http://zip4.usps.com.

[9] Based on Google Scholar: https://goo.gl/r4FaLh.

[10] http://dblp.uni-trier.de/db/.

[11] DM-Authors dataset: http://dx.doi.org/10.18709/PERSCIDO.2016.10.DS32.

[12] https://www.mturk.com/.

Author's personal copy

Table 3 Distribution of participants in user study

Age	Between 19 and 50 (med. 32)
Gender distribution	48% males and 51% females
Location distribution	53% USA, 33% India
Occupation	33% applied sciences

are deployed [31]. To remove "bias" from our study, we pick a task at random at each iteration of the user study [32]. As tasks are independent from each other, participants can rest between tasks to avoid the effect of "fatigue."

Summary of results Our user study shows that groups provide more actionable insights when compared to individual users. Analysts need a shorter time-to-insight to derive intuitions from groups (less than a minute in most cases). We also observe that groups enable a tighter decision-making loop by providing compact and aggregated pieces of information about users. We also compare diversity and coverage with other common interestingness measures in the literature and show that they are the preferred measures for group analytics. We also show that diversity highly correlates with informativeness (maximizing diversity leads less redundancy in results which enables more informative facets to appear), and coverage corresponds to representativeness (larger coverage leads a higher probability that almost all facets of the input users are represented.)

5.2.1 Group based versus user based analytics

In the first part of our user study, participants were given two pieces of information, side by side. The first piece is an exhaustive list of researchers (containing information about each researcher) and the second piece is the exhaustive list of groups (containing group labels). Participants were asked to mark at least "ten prolific young female researchers" (on DM-Authors dataset) in both lists. All groups contain at least 3 and at most 10 members for a fairer comparison with the user list. Participants can either scroll or perform a text search to reach users and groups of interest. Note that the task is not straightforward as being "prolific" is not defined in the dataset (see [30]), hence participants should check users and groups and decide if they satisfy the requested researchers.

We measure *time-to-insight* as one of the main key performance indicators in any data analytics solution. It is defined as the amount of time that participants need to find requested researchers in lists. Figure 5 illustrates the results. On average, participants needed 3.40 min on the group list and 6.37 min on the user list. We segment time-to-insight values into more understandable intervals in Data Science applications: 1 min is the safe time limit to keep track of the analyst's train of thoughts, while an analysis session of

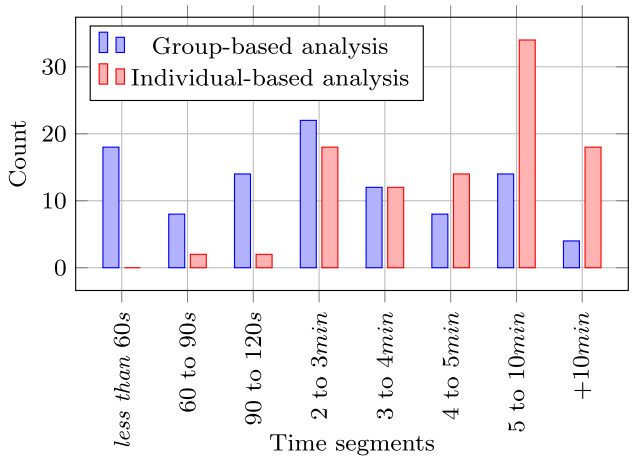


Fig. 5 Analysis of time-to-insight

10+ min indicates a serious interruption and unnecessary complication in the process [33]. We observe that the group-based task often terminates either in *less than a minute* or in *2–3* min. Conversely, the user-based task is mostly done in 5–10 min. We also note that absolutely no participant terminated a user-based task in less than a minute and only 4% performed in less than 2 min. Also note that the number of participants with a time-to-insight of 10+ min is one order of magnitude higher for the user-based case.

While we can easily conclude that group-based analytics means faster access to relevant user data, we also need to verify the amount of participants' mistakes for each approach (group-based vs user-based). In other words, we are interested to find out which approach is more useful by enabling participants to commit fewer errors in the analysis task. We associate an error rate between 0 and 100 to each participant which is proportional to the number of his/her missed and wrongly detected groups and users (according to the requested researchers). We observe that the average error rate (i.e., the lower the better) for the group-based case is 8.5 and for user-based is 20.7. A two-sample t test confirmed that the difference in error rates is significant. We conclude that group-based analytics is more advantageous with shorter time-to-insight.

We also ask participants to describe their experience of employing group and user lists in a free text. Some participants mentioned that they performed an easier and simpler "search" on groups rather than users, mainly thanks to a better organization and aggregation of user data in groups. They also noted that "group labels" enable early decision making. They also mentioned the following comments.

> *It is frustrating that each and every user in the user list should be verified, as there is no means to skip some. User list inspection would have been faster without the unrelated information. The presence of*

Author's personal copy

many (potentially uninteresting) attributes for each researcher complicates the decision-making process and makes the search action quite tricky.

The aforementioned comments highlight the importance of a group-based vision for user data analytics.

5.2.2 Interestingness measures

Among various interestingness measures discussed in the literature [10,34], we consider diversity and coverage as our global measures to obtain high-quality group sets both in GDISCOVER and GNAVIGATE. We passed two phases to validate our choice. First we performed a pilot study prior to our AMT study to compare various interestingness measures together. Once we obtained the majority of votes for diversity and coverage, we asked participants in AMT about their preference on groups generated with and without diversity and coverage. Note that in this experiment, we generate results with GDISCOVER. This is because we investigate on group sets in isolation (i.e., in connection with no other group set). Navigational aspects of UGA (i.e., GNAVIGATE) is discussed in Sects. 5.3.4 and 5.3.5 .

Pilot study In our pilot study, we recruited 35 local colleagues of our research laboratory, in person. We compare GDISCOVER group sets with the competitors generated using other interestingness measures. We ask participants to choose the most useful group set and justify their selection. Justi - cations can be J_1: understand who does what, J_2: discover new users and J_3: understand the whole data. The usefulness is judged based on the expressivity of group sets for the requested researchers. We consider four following interestingness measures as competitors (following the most common measures discussed in [10] and the most common employed in Data Science applications [35]):

- *Frequency* Optimizing frequency results in the largest groups possible (i.e., containing many group members).
- *Reliability* To be reliable, groups should con rm each other. Optimizing reliability results in most overlapping groups.
- *Novelty* Groups convey novel information if they don't repeat themselves. Optimizing novelty results in least overlapping groups.
- *Conciseness* Based on Occam's razor principle [36], the best choice is the one described with less information. A most concise group set has very short group labels.

Figure 6 illustrates the results. The left chart shows the average percentages of responses for each method and the right chart shows the justi cations. GDISCOVER dominance on other competitors is apparent by acquiring 40% of preferences. The second winner is conciseness with 25% of votes.

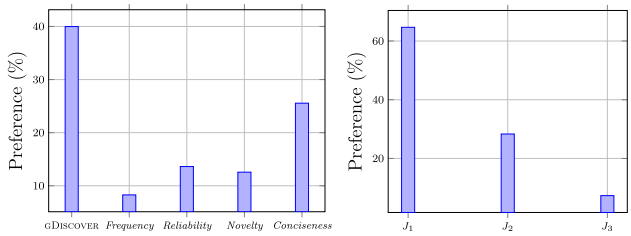


Fig. 6 Preference results for the pilot study on interestingness measures (left) and justi cations (right)

It reveals as a side note that beyond diversity and coverage, shorter descriptions influence better understanding of user data as well. We also observe that "too many overlaps" are annoying as most frequent groups are voted as the least useful. Also, our participants highly justi ed their choices as being "helpful to understand who does what," which certi es that optimizing diversity and coverage provides shortcuts for user behavioral analytics.

AMT User Study Our pilot study con rms our choice of interestingness measures. But one question is still unanswered: *"are coverage and diversity useful for user group analytics?"* For this study, we consider four baselines of GDISCOVER each of which generates a group set: DIVERSITY ONLY which maximizes diversity but not coverage, COVERAGE ONLY which maximizes coverage but not diversity, BOTH which maximizes both diversity and coverage (i.e., full functionality of GDISCOVER), and RANDOM which optimizes nothing and returns a random set of groups. For each group set generated by one the above baselines, we ask participants to provide a preference score (between 0 and 5) for two following measures which quantify the usefulness of the group set.

- **Representativeness** It reflects the extent to which the group set reflects the content of the input subset of users. The score 0 denotes the lowest representativeness and 5, the highest.
- **Informativeness** It reflects the extent to which the group set provides useful insights into the input subset of users. The score 0 means that no information is conveyed, 5 means being fully informative.

To better understand the difference between representativeness and informativeness, we consider an example subset of users described as "students of Computer Science in France." The following group set with two groups is highly representative: [student, studying Computer Science], [student, living in France, studying engineering]. However, it is not informative enough, as it contains redundant information and does not convey adequate novelty. On the other hand, the following group set is highly informative: [male,

Author's personal copy

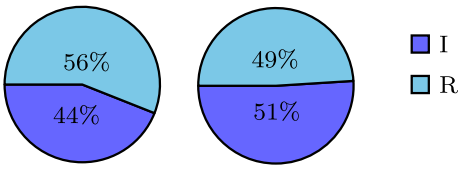


Fig. 7 Independent user study by reporting informativeness (I) and representativeness (R) for DIVERSITY ONLY (right), COVERAGE ONLY (middle) and BOTH (right)

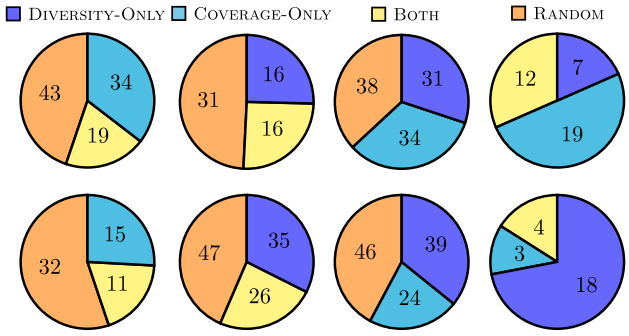


Fig. 8 Number of pairwise preferences for each baseline in the comparative study for informativeness (top) and representativeness (bottom)

teacher assistant student on Data Mining, studying in University of Paris-Sud], [female, student, living in south of France, studying Computer Science]. However, this group set is not representative enough as it is not descriptive for many computer science students in France. The following group set constitutes a good compromise between representativeness and informativeness: [student, living in France, studying Computer Engineering], [student, studying Computer Science]. Although there are still missing members of the input users who are not described with neither of the above groups, but a majority is captured.

The evaluation of our baselines consists of an *independent study* where each baseline is evaluated separately, and a *comparative study* where results of competitive methods are evaluated together.

Figure 7 illustrates the results for the independent study. The aim of the independent study is to check the quality of each baseline by itself. For each baseline, we show in percentage the score of representativeness and informativeness. While there is no signi cant difference between the scores in the case of BOTH (4.12 for informativeness and 3.92 for representativeness), there is a supremacy toward representativeness for COVERAGE ONLY (scoring 3.7 vs 2.9) and toward informativeness for DIVERSITY ONLY (scoring 4.2 vs 3.2). Diversity alone raises the score of informativeness by illustrating *various aspects* of the input users. However, it fails to fully represent the users, as the resulting group set may not express all of them. Also coverage raises the score of representativeness as its results hold *for most users*, but they may not be necessarily informative due to too many overlaps.

Our independent study resulted in many false positive and false negative results which we eventually removed. They were mainly due to participant's confusion between data attributes and the semantics of our measures. For instance, few participants thought that a group set with more underrepresented attribute values (e.g., females) becomes more representative. Few others thought that having longer group labels make a group set more informative. Some others thought that when they don't see an attribute in a group set (e.g., gender), it is not representative. To remove all these wrong assumptions, we attached explicit descriptions to the AMT task with illustrative examples.

In the comparative study, we show a pair of baselines at a time side by side, and ask the participants which one is more informative and representative. We then count the number of times each baseline is preferred to its competitor. Figure 8 illustrates the results. Numbers on the slices of pie charts show the *absolute* number of times (out of 50, i.e., the number of participants) that the method is preferred to its competitor. For instance, COVERAGE ONLY is preferred 35 times to DIVERSITY ONLY for representativeness. We observe following insights into our comparative study.

- Obviously, RANDOM has the lowest preference. The extreme case for informativeness is DIVERSITY ONLY vs. RANDOM where the former wins in 86% of cases. Also in representativeness, COVERAGE ONLY wins RANDOM in 94% of cases. Note that although RANDOM has chunky slices on the pie charts, but the numbers on them are low.
- For informativeness, the results are consistent with the independent study, where there is a bias in preference toward diversity. This nding is statistically signi cant as $F = 4.57 > F_{critical}$ where $F_{critical} = 2.71$ in ANOVA settings. DIVERSITY ONLY wins COVERAGE ONLY in 68% of cases. However, the most successful baseline for informativeness is BOTH which has the highest number against both DIVERSITY ONLY and COVERAGE ONLY with 62% and 68% of wins, respectively.
- For representativeness, the results are also consistent with the independent study, where there is a bias in preference toward coverage. This nding is statistically signi cant as $F = 3.13 > F_{critical}$ in ANOVA settings. In this case, DIVERSITY ONLY solutions are beaten in 70% of cases against COVERAGE ONLY. Still the winner is BOTH which bestows preferences of 78% and 48% to DIVERSITY ONLY and COVERAGE ONLY, respectively.

Figure 9 represents a summary of our comparative study where all preference counts for each baseline is summed up (illustrated in percentage). We observe that a good majority of

Author's personal copy

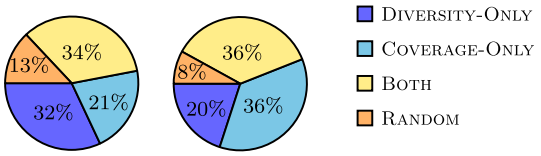


Fig. 9 Sum of preferences per baseline for informativeness (left) and representativeness (right)

preferences goes to DIVERSITY ONLY in case of informativeness, and to COVERAGE ONLY in case of representativeness. We conclude that "coverage" and "diversity" alone can capture a good amount of representativeness and informativeness, respectively. However, when combined, they absorb more preferences as both usefulness aspects are satis ed. BOTH generates group sets which represent the input users (by maximizing coverage) and convey maximal information (by maximizing diversity). This is why it owns at least one-third of preference votes for both informativeness and representativeness.

One interesting observation in our comparative study is that participants nd COVERAGE ONLY more compensative than DIVERSITY ONLY. Because in case of representativeness, COVERAGE ONLY is preferred as much as BOTH (i.e., 36%) and in case of informativeness, it can still gain 21% of preferences. When we analyzed free texts, we realized that for most participants "being representative" has a higher priority than "being informative." In other words, they prefer to rst make sure they have everything they need for their analysis task, and then think of informativeness. However, a combined approach proposed in UGA framework satis es both priorities at the same time.

5.3 Quantitative experiments

In this section, we evaluate the quantitative aspects of our UGA framework. Albeit we discussed the usability of our approach in Sect. 5.2, we still need to analyze the ef ciency of the UGA framework. Particularly, we are interested to address following concerns in UGA.

- **C1.** In both GDISCOVER and GNAVIGATE, there exists a parameter k which denotes the size of the output group set. What is a good value for k and how can an analyst tune k based on her analysis needs? How does k affect UGA's overall performance?
- **C2.** In GDISCOVER, why is it necessary to optimize objectives simultaneously (i.e., multi-objective optimization)?
- **C3.** Which one of the h-DISCOVER and -DISCOVER algorithms are faster? What are the influencing factors?
- **C4.** In GNAVIGATE, what does a "good" navigation on user groups mean? What factors do impact the fruitfulness of a navigation?
- **C5.** Does there exist a principled methodology to evaluate the need for an interactive multi-step user group navigation approach?

In the following sections, we discuss the aforementioned concerns in details. In **C1** to **C3**, we focus on MOVIELENS, because "user behavior" (i.e., voting) can be quanti ed with a rating score. As our experiments on other quanti ed behavior datasets (e.g., BOOKCROSSING) led to nearly identical results [23], here we only report MOVIELENS in the interest of space. For our quality-based experiments **C4** and **C5**, we employ DM AUTHORS to analyze user behaviors in more details.

Summary of results Through our extensive set of quantitative experiments, we show that groups with minimum size of 10 and group sets with size [3−5] are the best t for analyst consumption. We also observe that the multi-objective nature of GDISCOVER provides outstanding group sets where no quality dimension is sacri ced for others. We show that the approximation variant of GDISCOVER generates high-quality group sets, while the heuristic variant generates a subset in a more reasonable time. We also show that GNAV IGATE provides an effective navigation means which leads a knowledgeable analyst to successfully construct a program committee in 8 steps.

5.3.1 C1: group set size

It is shown in previous research [37] that offering at most 7 choices to an analyst matches perfectly her perception capacity. In this experiment, we vary k (number of groups in an output group set) between 2 and 10 and verify its influence on UGA. A smaller k means that the analyst receives smaller group sets in iterations of GNAVIGATE and in search results of GDISCOVER. We employ four different subsets of users on MOVIELENS: users voting for a movie with many ratings (i.e., American Beauty), few ratings (i.e., Celtic Pride), high rating scores (i.e., Kazaam) and low rating scores (i.e., Sanjuro). We want to verify if there is any correlation between the indicating factors of the user subsets and the parameter k.

Figure 10 illustrates the results. In the left chart, we plot the execution time of GDISCOVER to generate the exhaustive

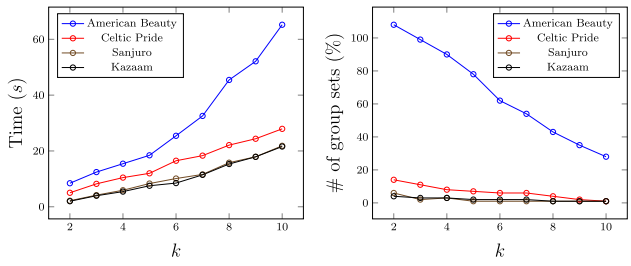


Fig. 10 Effect of k on execution time (left) and number of group sets (right)

Author's personal copy

set of all group sets. In the right chart, we show the percentage of materialized group sets comparing to all possible group sets. Note that we employ GDISCOVER in this experiment to evaluate the exhaustive behavior of UGA. GNAVIGATE's execution time subsumes GDISCOVER's. For all subsets of users, increasing k leads decreasing the size of the result space. Indeed, a bigger k means having larger group sets and less results. Nevertheless, for cases with less than 1000 users, the decrease is negligible. Also the execution time grows linearly with k. We often set k to values between 3 and 5 as they provide a good compromise between time (less that 10 s on average) and number of results (between 40 and 70% of all group sets).

5.3.2 C2: need for multi objective optimization in GDISCOVER

In GDISCOVER, we return group sets which have optimized values on coverage, diversity and diameter. But the principled question is *"what is the added value of multi-objective optimization?"* We first compare GDISCOVER with a single-objective optimization method and then discuss the relations between our objectives.

In the same context, one returned group set by GDISCOVER is the following: $G_{gDiscover} = \{g_4, g_5, g_6\}$ where g_4 = [female, young], g_5 = [young, living in Washington DC] and g_6 = [male, teenager]. The objective values for $G_{gDiscover}$ are as follows: $coverage(G_{gDiscover}, U) = 0.79$, $diversity(G_{gDiscover}, U) = 0.33$ and $diameter(G_{gDiscover}, U) = 0.11$. This group set has optimized values on all objectives. Specifically, it has a high diversity as only 2 female users are both young and residents of Washington DC. It also shows that min_c in MRI is a hard constraint and can easily miss a promising result which has a very high coverage but does not meet the threshold.

Consistency of objectives We already discussed that consistency of objectives transforms the multi-objective optimization problem into a simple single-objective optimization one which is trivial to solve (Sect. 3.6). In this experiment, we verify if our objectives (diversity, coverage and rating diameter) are consistent. We maximize coverage and observe how values of diversity and diameter evolve. To maximize coverage, we use Algorithm 3 discussed in Sect. 3.6.2. Figure 11 illustrates the results of four different input subset of users in MOVIELENS. Each point illustrates the objective values for each of 20 runs. We observe that in general, no correlation exists between the optimized value of coverage and other objectives, hence the need for simultaneous optimization of all quality dimensions.

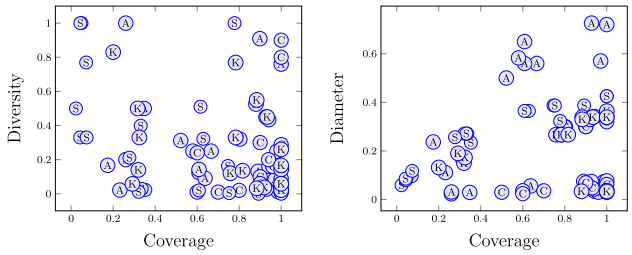


Fig. 11 Conflicting objectives in MOVIELENS. Movie title initials (**A**merican Beauty, **C**eltic Pride, **S**anjuro and **K**azaam) are illustrated on points

5.3.3 C3: comparison of GDISCOVER algorithms

The heuristic and approximation variants of GDISCOVER cover separate scopes of applications. -DISCOVER can be employed in an *archival* context to produce an exhaustive set of user group sets with a precision de ned by for further analysis. On the other hand, in a *streaming* context, h-DISCOVER is bene cial thanks to its *immediate* generation of a representative subset of results. In this experiment, however, we compare these algorithms in terms of ef ciency and quality of results. To conform with the state-of-the-art on evaluating multi-objective optimization approaches [38,39], we consider three different aspects for evaluating our two algorithms, -DISCOVER and h-DISCOVER: "cardinality," "diversity" and "coverage."

Cardinality based quality We employ ONVG[13] measure [38] to verify the number of solutions returned by -DISCOVER and h-DISCOVER. We consider 3 different instances for each algorithm: for -DISCOVER, we consider instances with = 2 (A), = 1.5 (B) and = 1.15 (C), and for h-DISCOVER, we consider instances with 5 (D), 10 (E) and 40 (F) diversity intervals. We run this experiment only with 4 largest subsets of users, as small subsets exhibit a similar predictable behavior [40]. As there is a direct correlation between the number of solutions and the execution time [22], we also report the performance of our algorithms.

Figure 12 illustrates the results of the ONVG study. As expected, in general the number of group sets generated by h-DISCOVER is one order of magnitude less than -DISCOVER. In both algorithms, the number of users plays an important role and increases the number of solutions. Also a data-centric observation in Fig. 12 reveals that more users lead more groups, hence worse performance (which is the case for the movie American Beauty).

Diversity based quality We examine the distribution and the extent of spread among the solutions of the Pareto front. We report Δ-metric [41] as the most accepted diversity-based

[13] Overall Non-dominated Vector Generation.

Author's personal copy

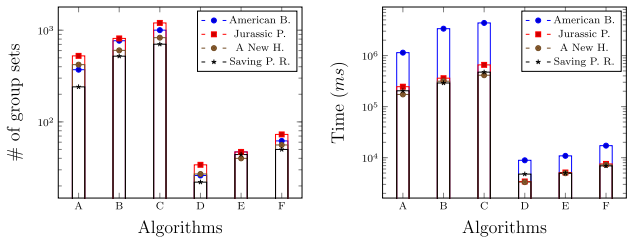


Fig. 12 ONVG quality comparison of -DISCOVER and h-DISCOVER (left) and resulting execution times (right)

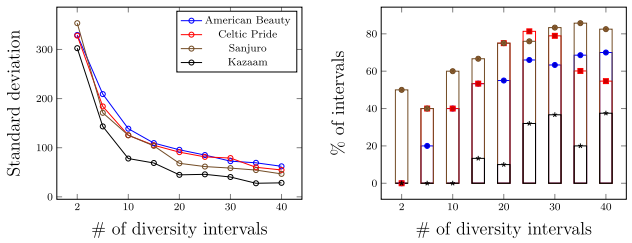


Fig. 13 Distribution and the extent of spread among group sets in diversity intervals

evaluation measure in the literature [38]. Figure 13 illustrates the results for different intervals and different input user subsets. The left chart illustrates the standard deviation for the number of group sets in intervals. If for an input subset of users, all intervals contain the same number of solutions (hence uniform), then the standard deviation is equal to zero, hence a better spread. Also, the right chart illustrates number of intervals with absolutely no solution, i.e., empty intervals. We observe a high heterogeneity when $n < 10$ for all input subsets of users. This means that by considering less than 10 intervals, we will potentially miss many Pareto plans. On the other hand, increasing the number of intervals leads to increasing the number of empty intervals which has the same consequence, i.e., missing Pareto plans. We then x n to 10 as it exhibits the best tradeoff between heterogeneity and emptiness. This value of n increases the chance of discovering more Pareto plans in h-DISCOVER, but as some amount of heterogeneity still remains even for n larger than 10, we cannot consider h-DISCOVER as a safe replacement for -DISCOVER.

Coverage based quality Beyond cardinality and distribution of results, we also verify the coverage of the objective space using the Pareto compliant $\mathcal{S}$-metric (aka, hypervolume) [42]. For each algorithm, we report the area covered between its generated solutions and a reference point, using normalized Lebesgue measure. The reference point is often picked as the least interesting point in the objective space, i.e., where all objectives (i.e., diversity, coverage and rating diameter) are equal to zero (assuming polarization is requested, hence maximizing diameter). Note that $\mathcal{S}$-metric is only usable when all the objectives are convex. This is the case for our three objectives as all of them have a nonnegative second-order derivative (see Eqs. 1, 2 and 3).

We employ the same subsets of users and same instances of algorithms that we used for cardinality-based evaluation, i.e., values of 1.15, 1.5 and 2 for -DISCOVER, and 5, 10 and 40 diversity intervals for h-DISCOVER. In general, we observe that the heuristic algorithm generates areas which are on average 22% smaller than the ones for the approximation algorithm. The largest area is obtained when $= 1.15$ for -DISCOVER (considering a normalized Lebesgue value of 1.0) and $n = 40$ for h-DISCOVER (having a normalized Lebesgue value of 0.73). Also for each separate algorithm, there is a linear growth in area by increasing the approximation area (i.e., decreasing) and increasing number of diversity intervals. It obviously shows that there is a direct correlation between the coverage area and the number of solutions. At the same time, the results of the heuristic algorithm exhibit a good compromise, as the shrinkage of its area is not drastic comparing to -DISCOVER.

Overall quality We report $\mathcal{C}$-metric [42] to perform an overall comparison between our two algorithms, -DISCOVER and h-DISCOVER. This comparison depicts the collective behavior of the methods regarding cardinality, distribution and coverage. For two multi-objective optimization algorithms X and Y, $\mathcal{C}(X, Y)$ measures the number of times that X's solutions are dominated by at least one solution of Y.

For our algorithms, we count the number of times each one dominates the other in pairwise comparison of their group sets. We consider $= 1.15$ for -DISCOVER and $n = 40$ for h-DISCOVER. We denote the set of -DISCOVER generated group sets as $\mathcal{P}$ and the set of h-DISCOVER generated group sets as $\mathcal{P}_h$. We observe that for all input subset of users in MOVIELENS (introduced in Sect. 5.3.1), at least 62% of group sets in $\mathcal{P}_h$ are dominated by $\mathcal{P}$'s. This is because -DISCOVER generates the complete set of -approximated Pareto plans, while h-DISCOVER materializes a subset. For instance, for the movie American Beauty, -DISCOVER generates 16 times more solutions than the heuristic algorithm.

Evidently the solutions in $\mathcal{P}_h$ are either as good as $\mathcal{P}$'s or worse. Concerning the huge difference in the size of solution sets, potentially a fairer comparison is to consider objective values to neutralize the influence of size. We observe that h-DISCOVER can achieve a supremacy over -DISCOVER in 39.4% of cases. This is a promising result for h-DISCOVER which is in line with our ndings regarding the dominance comparison.

5.3.4 C4: navigation quality

For GNAVIGATE, we are interested to measure *usability* and nd out what kind of navigational approach constitutes a "good" navigation. Then we validate whether GNAVIGATE

Author's personal copy

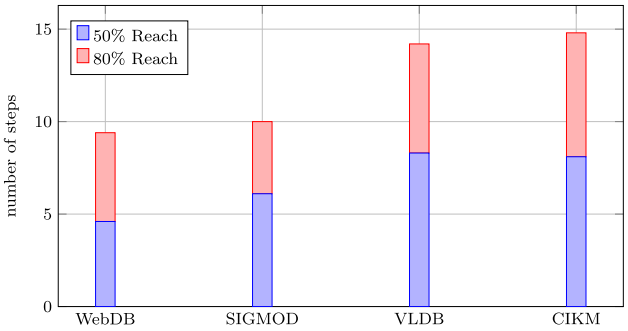


Fig. 14 Number of steps in GNAVIGATE for PC construction. The gure illustrates the results for the 2014 edition of the conferences. For VLDB, we only considered "review board" members. Also for CIKM, we only considered the "knowledge management" track

performs a fruitful navigation on user groups. In other words, we want to verify whether the sequence of group sets offered by GNAVIGATE appeals interesting to analysts. For this aim, we rst evaluate the overall process of navigation and then delve into influencing human factors.

Number of steps to reach target We verify the utility of our navigation component using a realistic example: building the program committee (PC) of major conferences/workshops in data management using DM AUTHORS dataset. For a given PC, we start from 5% of its members and use GNAVIGATE to nd the remaining ones. Target PC members should be found in user group sets proposed in different interactive steps of GNAVIGATE. Figure 14 reports the number of steps to discover 50% and 80% of PC members as the average of 50 runs of GNAVIGATE for each PC.

We can observe that any PC selection can be done in 12.04 steps on average. CIKM's PC is the hardest to discover and WebDB's the easiest. Our conjecture is that two key factors influence that: PC "size" and PC "diversity." Indeed, the PCs of VLDB, CIKM and SIGMOD contain over 100 members, while WebDB is smaller. This is why the former require a higher number of steps to cover 50% of their members (6.7, 6.5 and 5.9 steps respectively). In addition, the average pairwise Jaccard similarity (computed based on the pro le of researchers) between PC members of CIKM is 7.35. This high diversity results in more steps to reach 80% of their PCs (8.3 and 8.1 steps, respectively). SIGMOD has the least heterogeneous PC which leads to 4.8 steps to reach 80% of its PC. We also consider "disconnectedness," i.e., the average number of PC member pairs that have no attribute in common. We observe that there exist a direct relationship between diversity and disconnectedness, i.e., CIKM conference has also the highest disconnectedness score, i.e., 5.72 versus 0.48 for WebDB for instance.

Human factors in navigation We characterize different PC selection scenarios based on two human factors: *expertise* and *a priori knowledge*. To measure the effect of expertise, we consider two cases: a "knowledgeable" versus a "novice" PC chair. For the factor of a priori knowledge, we consider different starting points for the chair to build the PC: "a subset of the nal PC," "a subset of the previous year's PC," and a set of "arbitrary researchers outside the PC." We observe in general that GNAVIGATE can reach the target in 8 steps in case of a knowledgeable analyst. We also observe that a non-expert is more biased toward opExplore to discover the unknown space, while a knowledgeable chair uses both opExplore and opExploit. In the interest of space, we only review few scenarios.

In the KNOWIN scenario (Fig. 15), a knowledgeable analyst starts with a subset of the nal PC, i.e., George Fletcher, Martin Theobald, Sebastian Michel, and Xiaokui Xiao, and selects the last two as a seed group (because they are proli c young researchers with a high number of publications.) Exploring this group results in 3 groups out of which the one labeled with "SIGMOD" (the conference that hosts WEBDB) contains 4 researchers of interest (Lucian Popa, An-Hai Doan, Michael Benedikt and Sihem Amer-Yahia). The analyst then uses remove and then add actions to replace the predicate "high publications" with "data integration" (i.e., the WEBDB main theme in 2014) and decides to exploit the resulting group. In step D, the analyst keeps only Piero Fraternali and Felix Naumann among 12 group members using keep action. This action makes it easier to reach groups con-

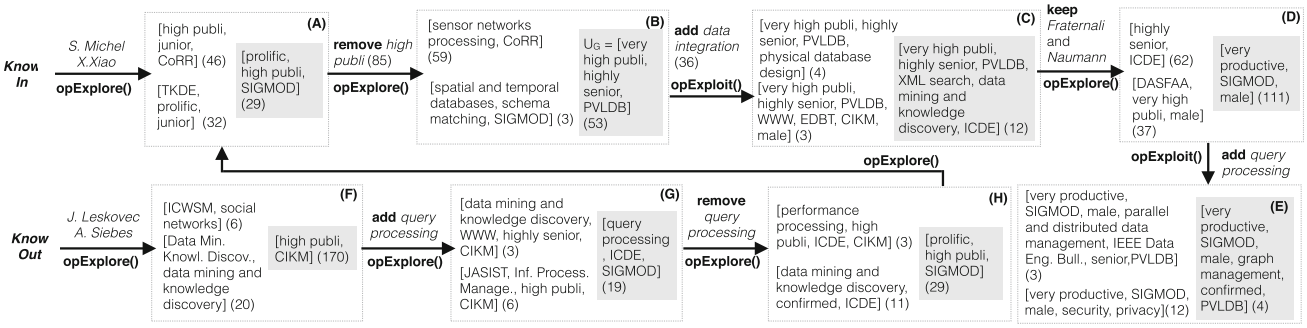


Fig. 15 Scenarios KNOWIN (top) and KNOWOUT (bottom) ($k = 3$)

Author's personal copy

taining items like "SIGMOD" (Piero Fraternali and Felix Naumann have 9 and 6 SIGMOD publications, respectively) and "ICDE" (e.g., Felix Naumann has 14 ICDE publications). Up to step E, the analyst is able to find 14 out of 15 PC members. The missing PC member is Jian Li. To understand why GNAVIGATE missed this member, we compare Li's activities with all other WEBDB 2014 PC members', and find that Li's research areas differ significantly from other PC members. This is a limitation of our approach where the consideration of relevance makes it hard to find users who are different from others.

In the KNOWOUT scenario, the knowledgeable analyst starts with Jure Leskovec and Arno Siebes, two researchers outside the final WEBDB PC. The opExplore operation first finds k-related groups that expand possible candidates. In step H, the analyst encounters the same group as in step A of scenario KNOWIN. This shows that in this case, a knowledgeable analyst only needs 2 more steps to reach relevant groups from a random departure point.

In scenarios with a junior PC chair (i.e., lack of expertise), we observe that the analysis is mostly done by opExplore. We also observe a tendency to manipulate group labels rather than group membership (specific researchers in groups). In case the junior chair starts with researchers outside the final PC, she repeatedly abandons a path and starts again with different groups.

5.3.5 C5: GNAVIGATE evaluation

Beyond the "collective" aspects of GNAVIGATE, we evaluate our navigation system for reaching a single target goal. For this experiment, we use a synthetic dataset which is generated to scale up MOVIELENS. It is a matrix M with 3×10^7 cells, where random squares with at least 10 users (i.e., $\sigma = 10$) are filled with 1, represent user groups. Then, we randomly mark 50 groups as targets denoted as $\mathcal{G}_{target}$.

We propose a measure called "Average Target Arrival" (ATA), i.e., the average number of iterations to reach a target group starting from a non-target group. We compare GNAVIGATE with three classes of baselines, namely UNSUPERVISED, OPTIMAL and NAVIGATIONAL. Briefly, if m_1 and m_2 are two different methods and $\text{ATA}(m_1) < \text{ATA}(m_2)$, then m_1 is considered faster and conceived a better option for navigation. Note that the concept of ATA differs significantly from finding the shortest path. For the latter, we assume the starting and target points are known, while this is not the case in a navigation process.

Algorithm 6 illustrates how ATA is computed. We designed 200 different sessions each of which has a different synthetic dataset and is repeated 100 times for each method. Hence, we compute 20,000 ATA values for each one of the baselines. For a random group g_{rnd}, k groups are returned using *method*, and a random choice between opExplore and opExploit (however, the algorithm starts always with an opExplore). Each of the k groups becomes the new seed. This depth-first recursive call terminates either when one group in $\mathcal{G}_{target}$ is found or when a path of length 50 has been built (the default value for *maxlength*). These recursive calls form paths inside the group space. A path is called *valid* if its last group belongs to $\mathcal{G}_{target}$. The ATA is computed as the average of valid path lengths for each method.

Algorithm 6: ATACompute algorithm

Input: $\mathcal{G}, \mathcal{G}_{target}, k, \mu, g, len, method, maxlen$
Output: length of navigation path
1 **if** $g \in \mathcal{G}_{target}$ **then return** len **if** $len > maxlen$ **then return** -1
// lost path $\mathcal{G}_k \leftarrow choose(opExplore(g, \mathcal{G}, \mu, k, method), opExploit(g, \mathcal{G}, k, method))$
2 **foreach** $g \in \mathcal{G}_k$ **do**
3 | ATACompute($\mathcal{G}, \mathcal{G}_{target}, k, \mu, g, len + 1, method, maxlen$)
4 **end**

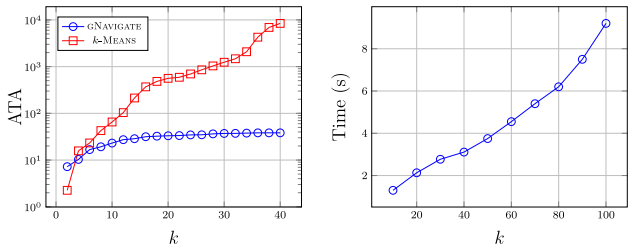


Fig. 16 Comparison of GNAVIGATE and UNSUPERVISED (left) and time for maximal diversity in GNAVIGATE (right)

Comparison with UNSUPERVISED baseline. We compare GNAVIGATE with a variant of k-MEANS (as a representative of clustering approaches) with "Jaccard" as the distance measure. At each step, both GNAVIGATE and k-MEANS return k groups while respecting the time limit. Any number of iterations is allowed for k-MEANS within *timelimit*. We then report ATA for both methods. For k-MEANS, we randomly add/remove attributes at each step i so that a new set of k clusters is obtained in step $i + 1$. The presence or absence of an attribute changes the clusters' membership, as the Jaccard distance between users varies. For instance, adding a specific value of the age attribute reduces the distance between two users having the same age.

Figure 16, left, illustrates ATAs for GNAVIGATE and k-MEANS in log scale. We vary k from 2 to 40 and observe how ATA for both algorithms evolves. While k-MEANS performs better for very small values of k, GNAVIGATE outperforms it by two orders of magnitude for higher values of k. When k is very small, clusters are huge. Thus, most of the time, there exists a cluster that contains all users of a target group. For larger values of k, more clusters with smaller size are generated and more steps are needed to finally reach the target. We can conclude that the superiority of GNAVIGATE over unsupervised methods comes from the use of diversity at each step in order to cover as many users as possible.

Author's personal copy

Table 4 Comparison with OPTIMAL baselines

	EXHAUSTIVE	ILP	GNAVIGATE
ATA	9.90	9.91	10.13
Time (s)	862.47	213.12	3.35

Comparison with OPTIMAL baselines We now compare GNAVIGATE with two optimal methods: EXHAUSTIVE and ILP. In each step of the navigation, EXHAUSTIVE generates all possible k among n groups in $\mathcal{G}$, i.e., $C\binom{n}{r}$ and chooses the one with the highest diversity. ILP returns k groups with maximal diversity using an integer linear programming formulation (using CHOCO 3.0 solver[14]). Optimal results are considered as the "gold standard" for ATA, as they function on the optimized value of diversity with no consideration of a time limit which leads the lowest possible ATA. Table 4 illustrates ATA and execution times for GNAVIGATE and optimal methods. Since both EXHAUSTIVE and ILP generate optimal paths, their ATA value is similar. However, their execution times differ. This experiment shows that GNAVIGATE is faster than optimal competitors (i.e., 3.49 min faster than ILP) while maintaining a comparable ATA.

Acknowledging the limitation of GNAVIGATE to obtain an optimal solution, we want to discover the amount of time needed to optimize diversity in GNAVIGATE regardless of the time limit parameter. Figure 16, right, illustrates the results. We observe that maximal diversity is achievable in less than 10 s even for large values of k. Note that main functionality of our *timelimit* parameter is to guarantee "continuity preserving latency" ($\sim$ 0.1 s) for navigation of user groups [43], i.e., the limit for the analyst to follow her train of thoughts instantaneously. If this tight guarantee is not required in an analysis task, GNAVIGATE can have an optimal functionality while still respecting "attention preserving latency" ($\sim$ 10 s).

Comparison with NAVIGATIONAL baselines We consider two navigational baselines, DIVRAND and PURERAND, which do not function on an optimized solution. At each step in the navigation, DIVRAND randomly generates as many sets of k groups as possible within *timelimit* and returns the one with the highest diversity. PURERAND, on the other hand, makes a pure random navigation regardless of the time limit. Figure 17, left, illustrates ATA results for NAVIGATIONAL baselines, by varying k from 2 to 40 (top), and varying number of groups from 50,000 to 1,000,000 (bottom). In general, we observe that GNAVIGATE has much lower ATA for $k \leq 16$ and $k \geq 30$. This simply shows that considering relevance and diversity at each step reduces ATA by an average of 15.91 steps.

For $k \in [16, 30]$, DIVRAND and GNAVIGATE have close results. This shows that although the *relevance* component

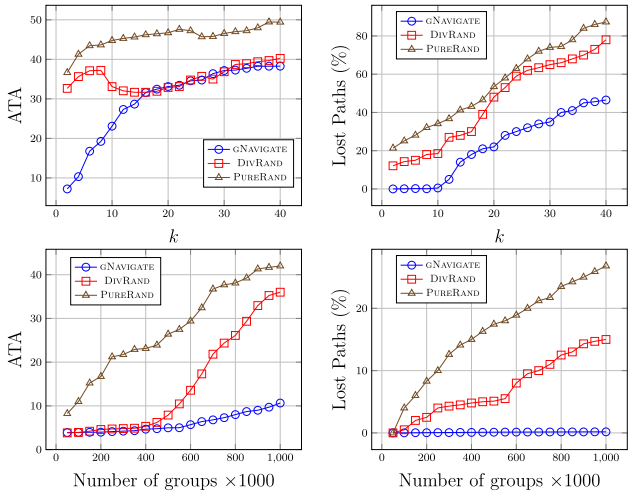


Fig. 17 Comparison with NAVIGATIONAL baselines

(i.e., the difference between DIVRAND and GNAVIGATE) is shown to be very useful in general, it is less effective for large values of k. In [44,45], it is shown that in a context with too many options and no hint for further navigation, "long jumps" are preferred to "short jumps." In our case, relevance tends to favor *short directed jumps* in the space of groups while DIVRAND does not. This is why when few options are available, GNAVIGATE performs better and DIVRAND performs as well as GNAVIGATE for larger values of k. We also observe that increasing the number of groups has a huge effect on DIVRAND. When the number of groups increases, the target groups are more likely to be diverse. Thus, precision (ratio of valid paths over all navigated paths) decreases for all methods, while thanks to relevance, the decrease is negligible for GNAVIGATE.

Figure 17, right, illustrates the percentage of lost paths for NAVIGATIONAL baselines, by varying k from 2 to 40 (top), and varying number of groups from 50,000 to 1,000,000 (bottom). As in line 1 of Algorithm 6, any navigation with more than 50 steps counts as a lost path. Our first observation is that GNAVIGATE generates very few lost paths comparing to its competitors. As k grows, more paths are generated and the number of lost paths grows accordingly. Our second observation is that the number of groups does affect the number of lost paths for GNAVIGATE. This is because of the fact that GNAVIGATE generates almost no lost path for lower values of k. Hence, with a low k, no lost path will be generated whatever the number of groups is.

6 Related work

We presented our UGA framework, a group-based analytical framework to understand user data. To the best of our

[14] http://choco-solver.org/?q=Choco3.

Author's personal copy

knowledge, the contributions that we presented for UGA framework have never been addressed in the literature. However, each contribution does relate to a number of others in its concept and functionality.

6.1 Hypothesis generation

We proposed GDISCOVER to materialize the exhaustive list of interesting user group sets by optimizing multiple objectives. Related work is considered on the aspects of "group discovery" and "multi-objective optimization."

Group discovery There is a recent trend in developing and reporting statistics about pre-de ned groups.[15] Group sets obtained by GDISCOVER are not static and can handle as many input user subsets as possible. Other dynamic discovery methods can be categorized to "attribute-based" and "action-based."

Attribute-based discovery Such discovery methods exploit user attributes (such as gender, age, occupation) and relations (such as friendship, af nity) for discovering groups of users. Social discovery is a body of work which employs user relations to form communities [46–50]. This means that the user data is divided into communities, such that users within the same community tend to be connected by links, while those within different communities tend not to be connected. Many networks are heterogeneous, consisting not of an undifferentiated mass of vertices, but of distinct groups. Our user group discovery problem differs in nature with community detection, because we do not consider any explicit relation between users in our data model. Representative discovery aims to mine groups that best represent a subset of users [51]. For instance in [52], an LSTM-based discovery method[16] is proposed to discover patients' subcohorts and facilitate therapeutic intervention. While most representative discovery approaches optimize a single objective at a time, GDISCOVER aims to discover representative user groups with optimized values on multiple conflicting objectives.

Action-based discovery The process of mining groups based on their common actions is called action-based discovery [53]. The most common objective in this category is *frequency*, which reports highly frequent regions in user data as groups [54]. More application-based objectives such as maximizing the return on investments are also proposed [55]. In UGA, we focus on multiple objectives tailored for user data. We build upon frequency and return high-quality groups with optimized values on coverage, diversity and rating diameter. On the other hand, most action-based methods discover non-overlapping clusters which is far from reality.

Team formation [56] and jury selection [57] are also new emerging action-based discovery methods. The main idea is to nd a group of experts to collectively complete a project. The focus is therefore on putting individual workers (users) together to optimize a quality objective, i.e., minimizing the overall cost while maintaining an acceptable overall expertise. The output of such algorithms is one single optimized group. Team formation requires the de nition of "quality" and "cost" values for each worker. This is a subjective and challenging task which requires the full knowledge of users' pro les in advance. On the other hand, the focus in our work is to generate a group set (not a single group) whose groups collectively optimize several objectives. In the case of team formation, the idea is to obtain only a single optimized group.

Multi objective Optimization There exists different approaches to solve a problem with a multi-objective nature. The prevailing approaches for solving multi-objective optimization problems in the state-of-the-art are evolutionary algorithms [58]. Instead of genetic-based mutations of evolutionary algorithms, we employ the dynamic programming approach in GDISCOVER. This allows us to bene t from nice mathematical properties of our objectives to prune the search space and reach real Pareto plans faster. Theorem 1 in Sect. 3.6.2 shows such property for coverage. In other words, we investigate on the special case of user data and a limited set of speci c objectives which best describe group sets. We provide formal de nitions of our objectives (coverage, diversity and rating diameter) and exploit their semantics to improve the ef ciency of our algorithms.

There exist other approaches which simplify the multi-objective optimization problem for nding near-Pareto solutions. We already discussed that Scalarization does not work in our case (Sect. 3.6). Another popular method is the ϵ-constraints method [59] where we optimize one objective and consider others as constraints. The approach in [51] can be seen as a relaxed ϵ-constraints version of our problem. Another approach is multi-level optimization [60] which needs a meaningful hierarchy between objectives. In our case, all objectives are independent and conflicting with each other, hence using this mechanism is not feasible.

For GDISCOVER, we adapt the dynamic programming approach in [22] and propose a multi-objective optimization algorithm for user group set discovery. The same idea is also proposed in [61] where Pareto plans are discovered entirely. However, an exhaustive approach to multi-objective optimization is very time-consuming. Thus, we propose an approximation algorithm, -DISCOVER, which is faster than exhaustive and provides bounds on the quality of results. We also propose a heuristic algorithm which returns a subset of -DISCOVER solutions in a reasonable time.

[15] http://blog.testmunk.com/how-teens-really-use-apps/.

[16] Long short-term memory networks.

Author's personal copy

6.2 Exploratory analysis

We proposed an interactive navigation approach for exploratory analysis of user groups. In an exploration scenario, the analyst only has a partial understanding of her needs and seeks to re ne them in iterative interactions. This awareness of analysts can be captured in different forms, such as queries, distributions, facets and examples. An interactive exploration system should employ the feedback received from the analyst to provide better results in consecutive steps. However, in most information navigation approaches in the literature, a lot remains to be done by the analyst, which puts burden on her. According to different ways of capturing the analyst's needs, we recognize the four following exploration types: by-query, by-analytics, by-facet and by-example. We briefly describe these exploration types and discuss why GNAVI GATE is implemented as a by-example exploration approach.

By-Query Interactions are formed using predicates on attributes and items which form a query. In each iteration of the analysis session, the analyst formulates a query and the exploration system returns groups which satisfy the query predicates [62–64]. Query formulation requires a knowledge of the dataset and the query language, which is not always the case.

By-Analytics Analysts explore the space of user groups using a desired distribution of members' actions. The exploration system returns groups whose distribution is similar to the input. For instance in [1], a desired histogram of rating scores is given and k groups with similar rating score distributions are returned. By-analytics exploration requires a knowledge of the user data and its underlying distributions.

By-Facet Another type of interaction is via attribute–value pairs (i.e., facets). The exploration system returns groups whose members satisfy requested facets [65,66]. Comparing to by-query exploration, by-facet exploration reduces the burden on the analyst speci cally on datasets with many attributes. However, the analyst needs to know possible facets in user data.

By-Example Interactions can also be made using examples. The analyst provides examples of what she needs to get and the system explores other groups similar to those provided examples. Example-based exploration is bene cial where the analyst is not able to express her needs otherwise [67]. We formulate GNAVIGATE as a by-example approach to provide the most intuitive way of exploring the space of user groups. In GNAVIGATE, we adopt an approach based on opExplore or opExploit operations and let the analyst choose which operation to apply at each step. However, the ability to "personalize" the navigation as in [44] is an interesting direction for future work.

7 Conclusion and perspectives

In this paper, we discuss group-based analysis of user data. We introduce UGA framework which puts together the components of user group analytics. Our framework serves two different use cases of user data analysis. First, our framework generates "interesting" group sets for hypothesis generation. Second, interesting group sets are populated on-the-go for exploratory analysis. We show that generating interesting group sets is an NP-Complete problem and propose an approximation and a heuristic algorithm for it. We also introduce GNAVIGATE, a group navigation method based on two navigational operations, opExplore and opExploit, which are both described as NP-Complete problems. We describe a greedy algorithm for GNAVIGATE which enables analysts to navigate in the space of groups and reach their interest. In an extensive set of experiments, we show the quality of generated user groups with our framework as well as it ef ciency against competitors.

Our immediate direction for future work is to enable visual exploration of user groups on the basis of the "visual information seeking mantra" [68]. The results of GDISCOVER and GNAVIGATE are not necessarily comprehensible and readable by analysts, unless a visualization layer enables sensemaking of user groups using visual variables. The combination of UGA's discovery and navigation with visualization, leads to visual analytics suits where analysts can interact with groups and perform their "what-if" scenarios in a human-understandable form. We plan to build upon our preliminary work [9] to achieve a full fledge automated pipeline of *discover-navigate-visualize* loop for user groups.

Appendix: NP hardness proofs

Theorem 2 *The decision version of* GDISCOVER *problem is NP-Complete.*

Proof It is shown in [51] that a single-objective optimization problem for user group set discovery is NP-Complete by a reduction from the Exact 3-Set Cover problem (EC3). There, homogeneity is maximized and a threshold on coverage is satis ed. In our case, two new conflicting dimensions (diversity and coverage) are added. This means that the problem in [51] is a special case of ours, hence our problem is obviously harder. □

For our proofs of hardness, we consider an in nite time limit in GNAVIGATE since that does not affect the complexity of our problem.

Theorem 3 *The exploration operation of* GNAVIGATE*, i.e., op xplore, is NP-complete.*

Author's personal copy

Proof The decision version of the problem is as follows: For a given group g, a set of groups $\mathcal{G}$ and a positive integer k, an overlap threshold μ, is there a subset of groups $\mathcal{G}' \subseteq explore(g, \mathcal{G}, \mu)$ such that (i) $g' \in \mathcal{G}' \wedge g' \neq g \wedge overlap(g, g') \geq \mu$ and (ii) $\Sigma_{(g_1,g_2)\in\mathcal{G}'|g_1\neq g_2}(1 - overlap(g_1, g_2))$ is maximized. A veri er v which returns true if both conditions (i) and (ii) are satis ed runs in polynomial time in the length of its input.

To verify NP-completeness, we reduce the MAXIMUM EDGE SUBGRAPH (MES) [69] (also known as DENSE K SUBGRAPH) to the decision version of our problem. The problem of MES is de ned as follows. Given an instance I consisting of a graph $G = (V, E)$, a weight function $w : E \rightarrow$, and a positive integer k, nd a subset $V' \subseteq V$, $|V'| = k$ such that the total weight of the edges induced by V', i.e., $\Sigma_{(v_i,v_j)}w(v_i, v_j)$ (where $(v_i, v_j) \in V' \times V'$) is maximized. This is an NP-complete problem [69] (originally reduced from the *Clique* problem).

Given I, we create an instance J of our problem as follows. J consists of a graph $G = (V, E)$ where the set of vertices $V = explore(g, \mathcal{G}, \mu)$ are groups that satisfy (i). Every pair of groups $(g_1, g_2) \in V \times V$ is also connected with a labeled edge, *i.e.*, $w(g_1, g_2) = 1 - overlap(g_1, g_2)$. The subset $V' \subseteq V$ ($|V'| = k$) is then a subset of groups where the sum of the weights between each pair of groups in V' is maximized, *i.e.*, $|E(V')| = \frac{k\times(k-1)}{2}$. The set V' is the most diverse subset of $\mathcal{G}$ that satis es the overlap condition ($\forall g' \in \mathcal{G}, overlap(g, g') \geq \mu$). Therefore, a set V' is a solution in instance I of MES *iff* it is a solution in instance J of our problem. Hence, the exploration problem is NP-complete. □

Theorem 4 *The exploitation operation of* GNAVIGATE*, i.e., op xploit, is NP-complete.*

Proof Similar to opExplore, a veri er v for exploitation runs in polynomial time in the length of its input. To verify NP-completeness, we reduce the MAXIMUM COVERAGE PROBLEM [70] to the decision version of our problem. The problem of MAXIMUM COVERAGE PROBLEM (MCP) is de ned as follows. Given an instance I consisting of m sets $S = \{ S_1 \ldots S_m \}$ where $S_i \in S_M$ (S_M being a reference set), and a positive integer k, nd a subset $S' \subseteq S$, such that $|S'| = k$ and the number of covered elements in S_M, i.e., $|\cup_{S_i\in S'} S_i|/|S_M|$ is maximized. This is an NP-complete problem [70]. Given I, we can create an instance J of our problem which consists of m sets $S = exploit(g, \mathcal{G}, \mu)$ and a reference group, i.e., $S_M = g_{in}$. In opExploit, we are interested to have k groups $S' \subseteq S$ that cover maximum number of users in S_M, i.e., $|\cup_{S_i\in S'} S_i|/|S_M|$ is maximized. Therefore, a set S' is a solution in instance I of MCP *iff* it is a solution in instance J of opExploit. Hence opExploit is NP-complete. □

References

1. Amer-Yahia, S., Kleisarchaki, S., Kolloju, N.K., Lakshmanan, L.V.S., Zamar, R.H.: Exploring rated datasets with rating maps. In: WWW (2017)
2. Amer-Yahia, S., Omidvar Tehrani, B., Roy, S.B., Shabib, N.: Group recommendation with temporal af nities. In: EDBT (2015)
3. Omidvar-Tehrani, B., Amer-Yahia, S., Termier, A.: Interactive user group analysis. In: CIKM (2015)
4. Cao, L.: Behavior informatics to discover behavior insight for active and tailored client management. In: SIGKDD (2017)
5. Wikipedia. Behavioral Analytics. https://en.wikipedia.org/wiki/behavioral_analytics (2014). Accessed 15 Mar 2018
6. Abiteboul, S., Bonchi, F., Oliver, N., Yu, B.: Toward personal knowledge bases. In: DSAA (2015)
7. Gramazio, C.C., Schloss, K.B., Laidlaw, D.H.: The relation between visualization size, grouping, and user performance. TVCG **20**, 1953 (2014)
8. Doodson, J., Gavin, J., Joiner, R.: Information seeking, acquainted with groups and individuals: information seeking, social uncertainty and social network sites. In: ICWSM (2013)
9. Amer-Yahia, S., Omidvar-Tehrani, B., Comba, J., Moreira, V., Zegarra, F.C.: Exploration of user groups invexus. In: ICDE demo (2018)
10. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: a survey. ACM Comput. Surv. (CSUR) **38**(3), 1–32 (2006)
11. Vreeken, J., Van Leeuwen, M., Siebes, A.: Krimp: mining itemsets that compress. Data Min. Knowl. Discov. **23**(1), 169–214 (2011)
12. Sidana, S., Mishra, S., Amer-Yahia, S., Clausel, M., Amini, M.-R.: Health monitoring on social media over time. In: SIGIR (2016)
13. Amer-Yahia, S., Rousset, M.-C.: Toppi: an ef cient algorithm for item-centric mining. In: DaWaK (2016)
14. Harper, F.M., Konstan, J.A.: The movielens datasets: history and context. ACM Trans. Interact. Intell. Syst. (TiiS) **5**, 19 (2016)
15. Bertin-Mahieux, T., Ellis, D.P.W., Whitman, B., Lamere, P.: The million song dataset. In: ISMIR (2011)
16. Monroe, M., Lan, R., Lee, H., Plaisant, C., Shneiderman, B.: Temporal event sequence simpli cation. TVCG **9**, 2227 (2013)
17. Ziegler, C.-N., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversi cation. In: WWW (2005)
18. Uno, T., Asai, T., Uchida, Y., Arimura, H.: Lcm: an ef cient algorithm for enumerating frequent closed item sets. In: Proceedings of Workshop on Frequent itemset Mining Implementations FIMI03 (2003)
19. Zhao, Z., De Stefani, L., Zgraggen, E., Binnig, C., Upfal, E., Kraska, T.: Controlling false discoveries during interactive data exploration. In: Proceedings of the 2017 ACM International Conference on Management of Data, pp. 527–540. ACM (2017)
20. Xu, C., Brown, S., Grant, C., Weaver, C.: Interactive visual analytics for Simpsons paradox detection. In: HILDA (2018)
21. Ganguly, S., Hasan, W., Krishnamurthy, R.: Query Optimization for Parallel Execution. ACM, New York (1992)
22. Trummer, I., Koch, C.: Approximation schemes for many-objective query optimization. In: SIGMOD. ACM (2014)
23. Omidvar-Tehrani, B., Amer-Yahia, S., Dutot, P.-F., Trystram, D.: Multi-objective group discovery on the social web. Research Report RR-LIG-052, LIG, Grenoble, France (2016)
24. Russell, S.J., Norvig, P.: Probabilistic reasoning. In: Arti cial Intelligence: A Modern Approach. Pearson Education Ltd (2003)
25. Robinson, D.J.S.: An Introduction to Abstract Algebra. Walter de Gruyter, Berlin (2003)
26. Liu, A.-A., Yu-Ting, S., Wei-Zhi, N., Kankanhalli, M.: Hierarchical clustering multi-task learning for joint human action grouping and

Author's personal copy

recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(1), 102–114 (2017)
27. Nandi, A., Jagadish, H.V.: Guided interaction: rethinking the query-result paradigm. In: Proceedings of the VLDB Endowment (2011)
28. Sarawagi, S., Sathe, G.: i3: intelligent, interactive investigation of OLAP data cubes. In: SIGMOD, vol. 29, p. 589. ACM (2000)
29. Indyk, P., Mahabadi, S., Mahdian, M., Mirrokni, V.S.: Composable core-sets for diversity and coverage maximization. In: SIGART (2014)
30. Omidvar-Tehrani, B., Amer-Yahia, S., Termier, A.: Interactive user group analysis. Research Report RR-LIG-048, LIG, Grenoble, France (2015)
31. Kittur, A., Chi, H., Suh, B.: Crowdsourcing user studies with mechanical turk. In: SIGCHI (2008)
32. Eickhoff, C.: Cognitive biases in crowdsourcing. In: WSDM (2018)
33. Nah, F.F.-H.: A study on tolerable waiting time: how long are web users willing to wait? Behav. Inf. Technol. **23**(3), 153–163 (2004)
34. Kirchgessner, M., Leroy, V., Amer-Yahia, S., Mishra, S.: Testing interestingness measures in practice: a large-scale analysis of buying patterns. In: DSAA (2016)
35. Mishra, S., Leroy, V., Amer-Yahia, S.: Colloquial region discovery for retail products: discovery and application. Int. J. Data Sci. Anal. **4**, 17 (2017)
36. Encyclopædia Britannica. Ockhams razor. Encyclopædia Britannica Online. Encyclopædia Britannica Inc, Chicago, IL (2009). Accessed 21 June 2009
37. Miller, G.: Human memory and the storage of information. IRE Trans. Inf. Theory **2**, 129 (1956)
38. Riquelme, N., Von Lücken, C., Baran, B.: Performance metrics in multi-objective optimization. In: CLEI. IEEE (2015)
39. Ke, L., Deb, K., Yao, X.: R-metric: evaluating the performance of preference-based evolutionary multi-objective optimization using reference points. IEEE Trans. Evol. Comput. (2017)
40. Omidvar-Tehrani, B., Amer-Yahia, S., Dutot, P.-F., Trystram, D.: Multi-objective group discovery on the social web. In: ECML/PKDD, pp. 296–312. Springer (2016)
41. Deb, K.: Multi-objective Optimization Using Evolutionary Algorithms, vol. 16. Wiley, New York (2001)
42. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)
43. Fekete, J.-D., Primet, R.: Progressive analytics: a computation paradigm for exploratory data analysis (2016). arXiv preprint arXiv:1607.05162
44. Boley, M., Kang, B., Tokmakov, P., Mampaey, M., Wrobel, S.: One click mining: interactive local pattern discovery through implicit preference and performance learning. IDEAS (ACM SIGKDD Workshop) (2013)
45. West, R., Leskovec, J.: Automatic versus human navigation in information networks. In: ICWSM (2012)
46. Mampaey, M., Tatti, N., Vreeken, J.: Tell me what i need to know: succinctly summarizing data with itemsets. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 573–581. ACM (2011)
47. Newman, M.E.J.: Detecting community structure in networks. Eur. Phys. J. B Condens. Matter Complex Syst. **38**(2), 321–330 (2004)
48. Yang, J., Leskovec, J.: Overlapping communities explain core–periphery organization of networks. In: Proceedings of the IEEE (2014)
49. Leskovec, J., Lang, K.J., Mahoney, M.: Empirical comparison of algorithms for network community detection. In: WWW (2010)
50. Cai, H., Zheng, V.W., Zhu, F., Chang, K.C.-C., Huang, Z.: From community detection to community pro ling. In: Proceedings of the VLDB Endowment (2017)
51. Das, M., Amer-Yahia, S., Das, Gautam, M., Yu, C.: Meaningful interpretations of collaborative ratings. In: VLDB (2011)
52. Baytas, I.M., Xiao, C., Zhang, X., Wang, F., Jain, A.K., Zhou, J.: Patient subtyping via time-aware LSTM networks. In: SIGKDD, pp. 65–74. ACM (2017)
53. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications, vol. 27. ACM (1998)
54. Srikant, R., Agrawal, R.: Mining generalized association rules. ACM (1995)
55. Pandey, S., Aly, M., Bagherjeiran, A., Hatch, A., Ciccolo, P., Ratnaparkhi, A., Zinkevich, M.: Learning to target: what works for behavioral targeting. In: CIKM (2011)
56. Kargar, M., An, A., Zihayat, M.: Ef cient bi-objective team formation in social networks. In: Machine Learning and Knowledge Discovery in Databases. Springer Berlin, Heidelberg (2012)
57. Cao, C.C., She, J., Tong, Y., Chen, L.: Whom to ask? Jury selection for decision making tasks on micro-blog services. VLDB **5**, 1495 (2012)
58. Coello Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-objective Problems, vol. 5. Springer, Berlin (2007)
59. Papadimitriou, C.H., Yannakakis, M.: On the approximability of trade-offs and optimal access of web sources. In: FOCS (2000)
60. Migdalas, A., Pardalos, P.M., Värbrand, P.: Multilevel Optimization: Algorithms and Applications. Springer, Berlin (1997)
61. Soulet, A., Raïssi, C., Plantevit, M., Cremilleux, B.: Mining dominant patterns in the sky. In: ICDM. IEEE (2011)
62. Bonchi, F., Giannotti, F., Lucchese, C., Orlando, S., Perego, R., Trasarti, R.: Conquest: a constraint-based querying system for exploratory pattern discovery. In: ICDE. IEEE (2006)
63. Bonchi, F., Giannotti, F., Mazzanti, A., Pedreschi, D.: Exante: anticipated data reduction in constrained pattern mining. In: PKDD, vol. 2838, pp. 59–70. Springer (2003)
64. Kifer, D., Bucila, C., Gehrke, J., White, W.: Dualminer: a dual-pruning algorithm for itemsets with constraints. In: SIGKDD (2002)
65. Yan, N., Li, C., Roy, S.B., Ramegowda, R., Das, G.: Facetedpedia: enabling query-dependent faceted search for wikipedia. In: CIKM (2010)
66. Khan, A.R., Garcia-Molina, H.: Crowddqs: dynamic question selection in crowdsourcing systems. In: Proceedings of the 2017 ACM International Conference on Management of Data. ACM (2017)
67. Mottin, D., Lissandrini, M., Velegrakis, Y., Palpanas, T.: New trends on exploratory methods for data analytics. Proc. VLDB Endow. **10**(12), 1977–1980 (2017)
68. Shneiderman, B.: The eyes have it: a task by data type taxonomy for information visualizations. In: The Craft of Information Visualization, pp. 364–371. Elsevier (2003)
69. Feige, U., Kortsarz, G., Peleg, D.: The dense k-subgraph problem. Algorithmica **29**(3), 410–421 (2001)
70. Johnson, D.S.: Approximation algorithms for combinatorial problems. In: Proceedings of the 5th Annual ACM Symposium on Theory of Computing (1973)