Sequence Recommendations for Groups: a Dynamic Approach to Balance Preferences

Sara Migliorini*, Elisa Quintarelli, Mauro Gambini, Alberto Belussi, Damiano Carra

Computer Science Department, University of Verona (Italy)

Abstract

We are living in the age of recommendations: it has been estimated that two-thirds of the films viewed on Netflix come from recommendations while the 35% of Amazon sales regard goods suggested to users. There are many factors to consider when providing a new suggestion: in addition to being useful, it should also be relevant and serendipitous, starting from historical data previously collected. In particular, the notion of context has to be considered since it induces some dynamic aspects in the definition of user preferences. The role of context becomes particularly important when we shift from single (myopic) suggestions to be provided to an individual user, to sequences of recommendations for groups of users. When the preferences of individual users are combined to define the preference of a new ephemeral group, dynamic contextual concerns have to be considered in order to provide the best possible experience and extend the group life, preventing the defection of some members because their preferences are not balanced. In this paper we introduce our proposal for producing sequences of recommendations for groups of users which is based on the Multi-Objective Simulated Annealing optimization technique and takes into account dynamic aspects. Moreover, we propose some strategies for extracting the required dynamic information from log data typically available and present the experimental results of the application of our approach in some real-world

*Corresponding author

Email address: sara.migliorini@univr.it (Sara Migliorini)

case studies.

Keywords: Recommendation, Contextual preferences, Group preferences, Optimization

1. Introduction

Recommender systems have become one of the key technologies developed and applied by major IT companies, such as Amazon, Netflix and Google. The general idea is that starting from a big dataset of available items, the system is able to provide useful suggestions to users based on their preferences and previous choices. Most prominent examples are the list of available video-on-demand movies and series, songs, courses, job advertisements, restaurants nearby the user, or products that can be purchased in online stores. In many cases, such huge catalogues, if not properly filtered and personalized, could become detrimental for users, rather than a precious resource.

Many recommender systems have been studied and developed in literature [1], originally such systems concentrated on the tastes and preferences of single users. However, since there exist many scenarios in which activities to be suggested are inherently social (e.g. going to the cinema, eating out or visiting a city), such systems had to evolve in order to consider groups of users in place of single ones [2, 3].

The transition from single users to **group of users** is not straightforward, many aspects have to be considered to determine the preferences of a group starting from individual tastes. A first distinction that has to be made is between persistent and ephemeral groups [4]. Persistent groups are those where members have a history of activities performed together, while ephemeral groups are constituted by users who are together for the first time. Traditionally in literature, the latter ones are the most interesting [5], since in this case the group preferences have to be properly derived from the individual preferences of its members and the available information about similar groups, if any. Conversely, persistent groups can be treated as individual users in the preference definition, the log of their activities can be collected and analyzed and thus, are a simplification of the ephemeral ones. Therefore, in this paper we concentrate on the most general case of ephemeral groups.

It should not be surprising that individual tastes and preferences could change based on the current **context** a user is interacting with [6]. A typical contextual feature is the time: a user could prefer to do some activities during the night w.r.t. daytime, or during the weekend w.r.t. the working days. However, being part of a group introduces another important dynamic factor, namely the individual tastes and preferences of a single user could change based on the group of people she is performing activities together with. It has been experimentally observed [7] that, when users are performing activities together inside a group, the satisfaction of the other members plays a central role. In order words, in a group, the balance of members' satisfaction is more important than individual pleasure [8, 9]. Throughout the paper, when we need to evaluate how well the system is able to respect the individual preferences of the members inside the group, we have to consider that also the possible evolution of the group composition can influence the individual preferences. It has also been observed that, if a member is likely to leave the group soon, the other members are more inclined to satisfy her tastes over theirs. A typical case is for instance the one of a family, composed by adults and kids, watching TV programs together during the evening: from the analysis of the past history of similar groups, we know that kids are more likely to leave the group first. Therefore, adults can comply to satisfy kids' preferences for a cartoon, because adults can be satisfied later when kids have left the group, while kids have no additional chances to be satisfied in the near future.

To summarize, even if online recommendations of the next activity, either for single users, or for groups, have been studied extensively in literature [1] [4], in this paper we emphasize the dynamic aspects related to the current context the user is acting in. The notion of context we are considering includes both temporal information, the current group composition and its possible evolution.

As for the evolution, the kind of recommendation considered so far in the literature could be defined as *myopic*, since it focuses on one activity at a time. Such an approach does not take into account that, by considering an extended period of time that includes different activities, the requests of users in a group could be satisfied in different ways. The recommendation of a **sequence of activities** has received little attention. For instance, the literature considers some scenarios such as the set of points of interest for tourists [10], [11] or the list of songs to listen [12], in which planning ahead the recommendation of a whole sequence – given some constraints, such as the available time – provides more flexibility. Proposing a complete sequence of activities in place of a single item at time, reduces the thinking time required by a myopic recommender system. In the case of a watching TV activity, it can reduce the channel surfing that users typically perform to find something interesting to watch next; or in case of a tourist trip, it can prevent discussions to decide the next thing to visit and can maximize the experience. In general, whenever a user or group has a limited time to spend in performing some activities, suggesting a complete sequence decreases the time wasted in selecting the best next activity.

Most of the works available in literature for sequence recommendations focus on a single user and do not consider groups (see the survey [12]). Conversely, in this paper we extend existing works by considering sequences of recommendations for groups of users. Moreover, we give emphasis on the fact that considering sequences of recommendations, in place of a single one, introduces a third element of dynamism, known as the issue of order. It has been experimentally demonstrated that, when a suggestion regards sequences of activities, the overall satisfaction may strongly depend on the order of the items, more than one would expect [13]. Therefore, it follows that the preference of a user or group w.r.t. a certain activity or item could depend on the previous items suggested inside the same sequence.

In producing a sequence of recommendations for groups of users, the satisfaction of the group is only one of the aspects that have to be considered. For instance, other aspects to be considered are the amount of time that the group can spend together, or the total budget at disposal, and so on. In this paper we propose an approach in which the satisfaction of the group is one of the functions to be optimized. More specifically, we propose to solve such problem as an optimization problem and to use the Multi-Objective Simulated Annealing (MOSA) as optimization heuristic [14] to explore the search space. The novelty w.r.t. the state of the art is the emphasis given to the dynamic aspects involved in the definition of the group preferences. As extensively discussed above, such dynamicity comes from two distinct factors: (i) when users act in a group instead of individually, their preferences depend on contextual attributes, where the group composition is one of them, (ii) when sequences of items are recommended in place of individual ones, the preferences of users depend also on the items already added to the sequence.

The dynamic aspects introduced so far, suggest that the recommendation engine has to be an online system which promptly reacts to the current context and situation. Moreover, the system has to efficiently respond to group requests at run-time. For these reasons, we propose a MapReduce implementation of the Multi-Objective Simulated Annealing that is able to rapidly provide suggestions to groups taking care of all the aspects described above. The proposed solution has been evaluated by using real-world application domains regarding the TV watching activity and the visiting of touristic POIs that are performed together by a group of users.

This paper is an extension of a previous contribution published as a conference paper [15]. More specifically, several extension points have been added w.r.t. the previous version, starting from the problem formulation to the implementation itself. Concerning the problem formulation, the definitions of individual and group preferences have been properly extended in order to accomodate all the mentioned dynamic aspects. Moreover, some hints about how to retrieve or reconstruct the necessary information from historical log data are also provided, allowing to effectively use our technique. Regarding the MapReduce implementation, we not only provide additional details, but we also discuss about its correctness and scalability. Finally, the experimentation has been extended to other application domains and a baseline comparison is also provided.

The remainder of the paper is organized as follows: Sec. 2 deeply illustrates the considered problem, Sect. 3 provides a formalization of the proposed recommendation system, while Sect. 4 discusses the details of the MapReduce implementation. In Sect. 5 the proposed approach is evaluated w.r.t. some real-world case scenarios and in Sect. 6 some alternative approaches already presented in literature are summarized. Finally, Sect. 7 concludes the work.

2. Context and Motivation

This section introduces in more details the paper justifications and contributions, taking as motivating example the TV domain. Anyway, the proposed approach is general enough to be applied to any practical application, as we will formalize in Sect. 3.2 and exemplify in Sect. 5, where different application domains are used in the experiments. In the following, we first summarize the items related to the dynamic aspects of the recommendation construction (identified as DX), then present the paper contributions (identified as OX).

D1: Individual preferences depend on the temporal context. As discussed in the introduction, individual preferences about items or activities can change during the same day or during the week. For instance, concerning the TV shows, some programs could be preferred during the daytime, while others could be considered more appropriate for the evening or the weekend. For this reason, the preference of a user towards a certain item is considered as a function of the temporal context. More specifically, based on the particular application domain, we can identify some time slots inside the same day (e.g., *daytime* or *night*), or we can distinguish between the days of the week (e.g., *weekdays* or *week-end*).

D2: Individual preferences depend on the group composition. Another aspect that could determine a change in the individual preferences is the fact that the user is performing activities alone or inside a group. A user can consider some activities more suitable when performed in group rather than individually; moreover, some activities could be judged more appropriate when the group is composed of certain kinds of members with respect to others. For instance, in case of TV shows, the preference of an adult w.r.t. cartoons could be greater when she is in presence of kids, rather than when she is alone or inside an adult group.

D3: Groups that evolve with time. The third element of dynamicity to keep in mind is related to the fact that the group composition could evolve with time. We say that a group has changed whenever its current composition type changes, e.g. from *adults with kids* the group can evolve into *adults*, no matter the number of members. In case the group members know that some of them will leave the group in the near future, they will be inclined to satisfy the outgoing members before their leave. Let us consider again the TV context, when a family group composed by adults and kids are watching TV together, if the parents know that the children will leave the group soon, they could decide to satisfy the children with their loved show, since they can watch their preferred show later when kids go to bed. Therefore, individual preferences have a dependence on the future evolution of the group (*forward dependence*).

D4: From myopic recommendations to sequences of recommendations. The last dynamic aspect to be considered is induced by the consideration of recommendations about sequences of items instead of individual ones. When we suggest multiple items or activities to be performed one after the other, what we have previously suggested could have an impact on the pleasure associated to the subsequent ones. For instance in [13] the authors state that for reaching an optimal satisfaction, news program has to ensure: (a) a good narrative flow (i.e., they should show topically related items together), (b) a mood consistency, namely it may be better to show items with similar mood together (e.g., viewers may not like seeing a sad item in the middle of two happy items), and (c) a strong ending, namely it may be better to end the sequence with a well-liked item, since viewers may remember the end of the sequence most. Conversely to point D3, in this case individual preferences have a dependence on the past (*backward dependence*).

O1: From individual preferences to group preferences. The first contribution provided by the paper is the definition of how the preferences of an ephemeral group can be derived from individual preferences. In doing this, we will keep in mind the dynamic aspects D1-D4 described above. We not only define how to combine individual preferences into group preferences taking care of dynamic aspects, but we also provide a methodology for extracting dynamic individual preferences from the log data commonly available about historical users' behaviour. Relatively to this objective and with reference to the contribution provided in the previous paper, in this work we introduce the following improvements: (a) thanks to D3 we know that groups evolve with time, so there could be some individuals that will leave the experience earlier than others without any chance to be satisfied in the future. (b) D4 is a completely new aspect introduced in the proposed methodology and it requires to define a measure of *transition preference* from an item i_1 to an item i_2. (c) The last extension point provided to this objective is the concept of *preference balancing*, which deals with the satisfaction of the individuals inside the group. In [7] the authors highlight that ethical concerns are usually taken into consideration in practical contexts; indeed, humans typically apply some strategies, like the Average Strategy, the Average Without Misery Strategy or the Least Misery Strategy, to take care about individual preferences and to avoid individual misery. In accordance with [8], in this paper we consider two measures of preference balancing: Min-Max ratio and Jain's metric, while the former emphasizes the gap between the least and the highest user preferences, the latter encourages the group members to achieve closer utilities. Despite these choices, other balancing metrics could be plugged in the methodology in place of the chosen ones. The main difference between the metrics used here and the ones applied in [8] is that they are based on dynamic preferences rather than static ones and they are computed on sequences instead of individual suggestions.

O2: Optimization problem with objective functions and constraints. In this paper we consider the problem of providing sequences of suggestions for group of users as a multi-objective optimization problem. More specifically, we use the MOSA (Multi-Objective Simulated Annealing) heuristic for exploring the search space by taking into account various constraints to be satisfied and functions to be optimized. Different constraints and optimization criteria can be considered, in particular we identify six objective functions: (i) the minimization of the time interval between two consecutive items, (ii) the maximization of the portion of T_{max} covered by the recommendation, (iii) the minimization of the number of items the group members have already enjoyed in the past, (iv) the maximization of the group satisfaction, (v) the minimization of the gap between the least and the highest utilities, and (vi) the maximization of the closeness between user utilities. At the same time two constraints have been defined: the maximum interval of time the group can spend together (T_{max}) and the maximum available budget (b_{max}).

Serendipity. Recommender systems traditionally use past behaviours of users to suggest items. The idea is to satisfy the user's tastes and preferences by proposing something similar to what the user has indicated as interesting. The technique proposed in this paper uses a similar idea in the construction of the dynamic users' and groups' preferences. However, as recognized in literature, sometimes the users become bored with obvious suggestions that they might have already discovered. Therefore, to improve users' satisfaction, recommender systems introduce a degree of serendipity to the provided suggestions [16]. Serendipity is defined as the faculty of making fortunate discoveries by accident. The technique proposed in this paper tries to address this additional feature in two ways: (i) by taking care of the previous experiences performed by the same users through a specific objective function, (ii) by the MOSA itself which, during the initial step of the search space exploration, considers also worse solutions in order to not stuck in local optima.

Computational aspects. Finally, but equally important, the proposed solution should be efficient, since recommendations must be generated at runtime, when specific (groups of) users require them. The system needs to explore a huge solution space, and even using well known heuristics for solving optimization problems, the computational complexity still remains high. For this reason, the proposed solution should be implemented with an approach that allows for parallel computations, so the time necessary to produce the recommendations is kept as short as possible. The definition of a MapReduce version of the MOSA algorithm could be considered a contribution alone, since many other contexts different from recommender systems can benefit from it. With respect to our previous work, we not only provide more details, but we also include a study about its correctness and scalability.

3. A Recommendation System for Groups

This section provides a detailed description of the proposed recommendation system. In particular, we initially provide an overview of the solution architecture (Sect. 3.1) and then we formalize the considered problem (Sect. 3.2)

3.1. System overview

Figure 1 shows an overall picture of our approach: the three central rectangles identify pre-processing steps that are performed on the available logs in order to identify contextual dynamic preferences and possible group evolutions. Conversely, the violet rectangle, labelled as **MOSA**, represents the optimization procedure used for producing sequences of suggestions for dynamic groups.

The pre-processing steps can be performed off-line and the obtained results can be stored to make them efficiently available during the online activity. More specifically, starting from the past activities performed by individual users alone, it is possible to extract the dynamic individual preferences [D1][D2]. Conversely, by analysing the past sequences of activities performed by groups of users, we can identify the possible group type evolutions [D3]. As we will extensively discuss in the following sections, we are not interested in the group composition but only in its type. Therefore, evolution behaviours observed in groups can be extended to other groups having a different composition but an equal type (i.e.,

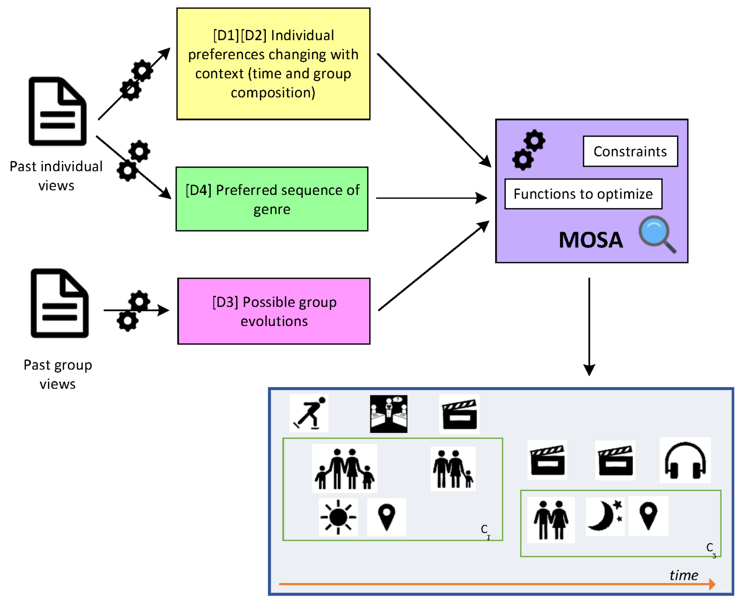

Figure 1: Our approach: a high level vision

management of ephemeral groups). Finally, by considering the past sequences of views, we can determine the preferred genre/type transitions that happen in a particular context [D4].

The results produced by these pre-processing steps are: (1) a set of dynamic individual preferences assigning to each pair composed of a user u and a genre gen, a preference value which depends both on the time and the group composition, i.e. on the context c: $D_{1,2} = \{\langle u, \mathsf{gen}, c, \mathsf{pref}\rangle\}$. (2) A dataset containing for each pair of genres gen_1 and gen_2 a value representing the preference associated to a transition inside a sequence from gen_1 to gen_2 in a given context c: $D_3 = \{\langle \mathsf{gen}_1, \mathsf{gen}_2, c, \mathsf{pref}\rangle\}$. (3) A dataset containing for each pair of group types ρ_1 and ρ_2 a value representing the probability that a group g changes its type from ρ_1 to ρ_2 in a context c: $D_4 = \{\rho_1, \rho_2, c, \mathsf{pref}\}$, where the notion of group type will be clearer in the following.

All these sets of information are used as input for the MOSA technique together with some detailed information about the currently available items (e.g., the current TV schedule or the POI timetable) and the various optimization criteria described in the following section. The present solution and the one in [15] share the same main architecture depicted in Fig. 1. Anyway, the implementation of each individual step has been properly extended and enriched. Sect. 3.2 discusses how the problem can be formulated as an optimization problem and then Sect. 4 shows our solution based on the MOSA technique.

3.2. Problem Formulation

This section formalizes the problem of producing a sequence of recommendations for a group of users in a given context. To better follow the formalization and support the reader, Tab. 1 summarizes the introduced symbols with their meaning. We consider as subject a set of entertainment activities defined as follows.

Definition 1 (Entertainment)**.** *An entertainment e is a leisure activity performed by a user or a group of users. It is characterized by several attributes, such as a duration $e.\mathsf{dur}$, a genre or type $e.\mathsf{gen}$ and a cost $e.\mathsf{cost}$.*

Notice that for certain kinds of entertainments, such as a museum visit, $e.\mathsf{dur}$ may refer to the suggested time to spend in the activity. Independently from the duration, a user (or group) can enjoy an entertainment e for an interval of time not equal to $e.\mathsf{dur}$, thus, we use the notation $e.\mathsf{start}$ and $e.\mathsf{end}$ to indicate when the user or group started and finished to enjoy e.

In this paper, we consider two application domains: the TV on-demand entertainments and the POI touristic visits. In both cases, each user can choose between as set of different shows or places that are characterized by a duration or a visiting time, a genre (e.g., documentary, sport, horror, action, etc) or type (e.g., churches, historical places, playgrounds, etc.) and possibly a cost for the viewing (e.g., some shows can be offered as a pay-per-view or included into a subscription offer) or the entrance. Clearly, independently from the show or activity duration, a user or a group can start to watch it after its beginning or can end the vision before its completion, or similarly they can enjoy the activity for a different period of time depending on the time at their disposal.

Symbol	Alt. Symbol	Meaning
$e \in \mathcal{O}$		entertainment
$\mathcal{T}$		set of entertainment types: $\{e.\mathsf{gen} \mid e \in \mathcal{O}\}$
$\varepsilon \in \mathcal{E}$		experience
$u \in \mathcal{U}$		user
$g(t)$	g_t	dynamic group at time instant t
$\rho(g(t))$	$\rho(g_t)$	dynamic group type at time instant t
$\tau(t)$		temporal characterization
$c(g_t) = \langle \rho(g_t), \tau(t) \rangle$		dynamic context
$\Delta(g_{t_i}, t_{i+1})$		set of possible evolutions of the group g_{t_i}
$\sigma(e_i.\mathsf{gen}, e_j.\mathsf{gen})$		transition preference from two genres/types
$\bar{p}(u, c(g_t), \varepsilon)$		preference of the user u in the context $c(g_t)$
$\bar{p}(g_t, c(g_t), \varepsilon)$		preference of the group g_t in the context $c(g_t)$
$r_m(g_t, c(g_t), \varepsilon)$		min-max ratio
$r_j(g_t, c(g_t), \varepsilon)$		Jain's metric

Table 1: Summarization of the symbols used in the problem formalization.

Definition 2 (Experience)**.** *An experience ε is an ordered collection of entertainments $\varepsilon = \langle e_1, \ldots, e_n \rangle$, where n indicates the number of entertainments contained in ε ($|\varepsilon| = n$).*

Given a set of possible entertainments $\mathcal{O}$, the set of all possible experiences, denoted by $\mathcal{E}$, contains all possible ordered combinations of entertainments in $\mathcal{O}$, for any cardinality of ε. In the considered domains, the set $\mathcal{O}$ is given by the shows available in the platform or the POIs of a given tourist location, while the set $\mathcal{E}$ contains all the possible sequences of entertainments built starting from $\mathcal{O}$. The construction of the set $\mathcal{E}$ will take care not only of the duration of each single element in $\mathcal{O}$, but also of the timeframe in which the show is put on air or the POI opening time. For instance, the vision for sport shows is usually performed live rather than delayed.

Let $\varepsilon = \langle e_1, \ldots, e_n \rangle$ be an experience, its overall duration $\delta(\varepsilon)$ is defined as

$$\delta(\varepsilon) = \sum_{i=1}^{n} e_i.\mathsf{dur} \tag{1}$$

while its overall cost $\gamma(\varepsilon)$ is defined as

$$\gamma(\varepsilon) = \sum_{i=1}^{n} e_i.\mathsf{cost} \tag{2}$$

and finally its global genre or type $\lambda(\varepsilon)$ is defined as

$$\lambda(\varepsilon) = \bigcup_{i=1}^{n} e_i.\mathsf{gen} \tag{3}$$

Let us consider for instance an experience ε composed by three entertainments e_1, e_2 and e_3, such that: $e_1 = \langle \mathsf{dur} : 1\,h, \mathsf{cost} : 4\$, \mathsf{gen} : \text{“commedy”}\rangle$, $e_2 = \langle \mathsf{dur} : 30\,min, \mathsf{cost} : 0\$, \mathsf{gen} : \text{“documentary”}\rangle$, $e_3 = \langle \mathsf{dur} : 15\,min, \mathsf{cost} : 1\$, \mathsf{gen} : \text{“cartoon”}\rangle$. From the previous equations it follows that $\varepsilon.\mathsf{dur} = 1\,h\,45\,min$, $\varepsilon.cost = 5\$$ and $\varepsilon.\mathsf{gen} = \{\text{“commedy”}, \text{“documentary”}, \text{“cartoon”}\}$.

Definition 3 (User)**.** *A user u is an individual performing some activities and is characterized by a set of properties, such as a type $u.\mathsf{type}$. In the following the set of all users is denoted as $\mathcal{U}$.*

The previous definition of “user” could be enriched with additional properties that depend on the considered application context. In this paper, for keeping the notation simple, we mention only the type since it is essential for the following definitions. Clearly, also the notion of type depends on the considered domain. For instance, we can classify users based on their age into “kid”, “teenager”, “adult”, “senior”; or based on their education level into “no education”, “elementary”, “high school”, and “higher”.

3.2.1. Dynamic Group Composition and Dynamic Context Definition

A set of users performing activities together forms a group. In this paper, we consider a dynamic notion of group whose composition evolves over time.

Definition 4 (Dynamic Group)**.** *Given a set of users $\mathcal{U}$, a dynamic group $g : \mathbb{I} \rightarrow \wp(\mathcal{U})$ is a function that associates to each time instant $t \in \mathbb{I}$ a subset of users $\{u_1, \ldots, u_n\} \subseteq \mathcal{U}$. Each (dynamic) group is characterized by a type ρ, which is dynamic too, thus it depends on the group composition at time t.*

$$\rho(g(t)) = \{u_i.\textsf{\textit{type}} \mid u_i \in g(t)\} \tag{4}$$

In the following for not cluttering the notation, we will denote $g(t)$ as g_t and $\rho(g(t))$ as $\rho(g_t)$. The group type $\rho(g_t)$ is essential for defining the notion of context considered in this paper. Indeed, we are interested in the group type for determining the context (e.g. *adults with kids* or *adults*), not in its specific composition, namely the individual users that are currently inside the group. This implies that the notion of context is dynamic too, but it does not strictly change with the group composition, rather it depends on the group type. Let us consider for example a family group composed of two parents and two children, i.e. the group type at instant t_i is $\rho(g_{t_i}) = \{\textsf{adult}, \textsf{teenager}\}$, if at instant t_{i+1} one of the two children leaves the group, the group type $\rho(g_{t_{i+1}})$ does not change even if the group composition has been changed. Conversely, if the group type at time t_i is $\rho(g_{t_i}) = \{\textsf{adult}, \textsf{teenager}, \textsf{baby}\}$ and at time t_{i+1} becomes $\rho(g_{t_{i+1}}) = \{\textsf{adult}, \textsf{teenager}\}$, then a group type change occurs.

Besides to the group type, another factor that determines the context is a temporal characterization of the period during which the group is performing activities together. More specifically, given a time instant t, we define a function τ which returns one or more labels characterizing it, such as "daytime" or "night", or the day of the week and so on. This temporal characterization together with the group type determines the notion of dynamic context.

Definition 5 (Dynamic Context)**.** *Given a dynamic group g_t, a dynamic context $c(g_t) = \langle \rho(g_t), \tau(t) \rangle$ for a group g at time t is given by its type $\rho(g_t)$ (e.g. adults or adults with kids) and a temporal information $\tau(t)$ (e.g. daytime or night, and the day of the week).*

The temporal information τ is essentially a compact representation of the date and time characterizing the instant t. For instance, we can identify different contexts based on several time slots in a day, or different days in a week, and so on. The general idea is that users can change their preferences during the weekend or during the evening, even if they are spending time with the same types of people. The definition of the function τ greatly depends on the considered application domain. Clearly, data analysts can provide an educated guess based on their experience with a specific application domain.

Given the notion of dynamic group and dynamic context, the next sections discuss how to build the preference of a group starting from the preferences of its members, considering also the possible evolutions of the group composition.

3.2.2. Individual Contextual Preferences

This section formalizes the notion of individual contextual preference and it describes how such concept can be computed starting from the log data usually collected about past user experiences. This last aspect could seem straightforward, but in many practical cases not all the information required by the proposed technique are directly available in the data, so the definition of such derivation formulas can increase the applicability of the technique in many real-world scenarios. In the definition of individual preferences, the dynamic aspects enter in the question in two ways: (i) by changing the preferences of the single user w.r.t. the notion of context (i.e., **contextual preferences**), (ii) by changing the preferences of the single user w.r.t. the items previously added to the sequence of suggestions (i.e., **sequence of activities**). Relating to this second aspect, as already discussed at the beginning of this paper, we have also to consider the so called "issue of order" [13]: the impact on the user's happiness in viewing an item or performing an activity is likely to depend not only on the preferences associated to the single element in isolation, but also on the user's mood induced by the previous activities. All these dynamic aspects are captured by the following definition of user preference.

Definition 6 (User Preference)**.** *Let* $c(g_t) = \langle \rho(g_t), \tau(t) \rangle$ *be a dynamic context for a group* g_t *at temporal instant* t*,* $\varepsilon = \langle e_1, \ldots, e_n \rangle$ *a partial experience* *built so far, and e_j be an entertainment. For each user $u \in g_t$ the function $\bar{p}(u, c(g_t), \varepsilon, e_j)$ computes the preferences of the user u in the context $c(g_t)$ in going on with the experience ε and enjoying the additional entertainment e_j.*

From Def. 6 it is clear that the individual preference of each user is dynamic in two ways: (i) it depends on the context $c(g_t)$ which is dynamic too, and (ii) it depends on the partial sequence built so far, namely the mood induced by the previous elements in the sequence influences the satisfaction of the user w.r.t. the next element in the sequence.

In the ideal situation, all the information regarding the preferences of users in any considered context are available. Unfortunately, in real situations, all these kinds of information are rarely available or are only partially available. In order to overcome this situation and extend the applicability of the proposed approach, the following two propositions describe how the contextual preferences of a user towards a certain entertainment can be derived from the commonly available activity logs. Firstly, Prop. 7 derives preferences towards a single entertainment activity, while Prop. 10 extends them to a sequence of activities.

Proposition 7 (From logs to myopic user preferences)**.** *Let us consider a set of records R containing tuples of the form $\langle u, g_t, start, end, e\rangle$, stating that the user u enjoined the entertainment e from start to end while she was inside the group g_t. From this log it is possible to derive the user preference described in Def. 6 in the following way: given a dynamic context $c(g_t) = \langle \rho(g_t), \tau(t)\rangle$, a user u and a genre gn, we identify the set of records:*

$$R_{u,c(g_t),gn} = \{\langle u_s, g_r, start, end, e\rangle \mid u = u_s \ \wedge \ \rho(g_t) = \rho(g_r) \wedge \\ t \geq start \wedge t \leq end \ \wedge e.\mathsf{gen} = gn\} \tag{5}$$

which regards activities of type gn performed by the user u in the context $c(g_t)$. We also identify a set $R_{u,c(g_t),*}$ as in Eq. 5 but without considering the condition on the activity type. Given these two sets, the preference $\bar{p}(u, c(g_t), e)$ of a user *u in a context $c(g_t)$ towards an entertainment e can be computed as:*

$$\bar{p}(u, c(g_t), e) = \frac{\sum_{r \in R_{u,c(g_t),e.\mathsf{gen}}} r.end - r.start}{\sum_{r \in R_{u,c(g_t),*}} r.end - r.start} \cdot \frac{1}{\#views_{\mathsf{gen}}} \tag{6}$$

In other words given the context $c(g_t)$, the preference of the user u for the entertainment e is computed as the ratio between the sum of the durations of activities of type $e.\mathsf{gen}$ performed by u in context $c(g_t)$ and the sum of the durations of all activities performed by u in the same context, independently from their type. Clearly, if few data are available for a particular genre, namely the global duration is very limited, the preference for this kind of entertainment is set equal to zero. However, we can distinguish two different cases: (i) the user completely dislikes such genre, (ii) the user has rarely tried a particular genre. The first case can be easily recognized because there are several very short views (i.e., the user starts an entertainment and abandons it very soon), in the second case there are only few views but with a considerable duration (i.e., the user rarely chose the genre in the past, but when it happened, the activity has been completed). In order to take care of some serendipity aspects, this second kind of situation has to be preferred in place of the first one. Therefore, in the computation of the myopic user preferences, besides to the durations, we also consider the number of times the genre has been chosen ($\#view_{\mathsf{gen}}$). Eq. 6 is a further modification of the Term Frequency-Inverse Document Frequency [17] adapted to our scenario in a preliminary way in [18]. More specifically, in [18] the authors consider only the number of times the movies of a genre i are chosen by a user u w.r.t. the total number of views performed by the same user u. Conversely, here we adapt the formula to accomodate the notion of context, but we also consider as relevant the amount of time spent in a given activity, instead of the number of times it has been chosen. This can be a more realistic measure about the pleasure of the user u w.r.t. an entertainment of type i.

Given the myopic individual preference in Eq. 6, in order to compute the user preference $\bar{p}(u, c(g_t), \varepsilon, e_j)$ in Def. 6, it is necessary to define a transition preference between two genres or activity types.

Definition 8 (Entertainment transition preference)**.** *Let us denote as $\mathcal{T}$ the set of entertainment types in the considered domain, we define a function $\sigma : \mathcal{T} \times \mathcal{T} \rightarrow \mathbb{R}$ that given two entertainments $e_i, e_j \in \mathcal{E}$ returns a value representing a measure of preference for a transition from the type $e_i.\mathsf{gen}$ to the type $e_j.\mathsf{gen}$.*

Notice that the notion of transition preference has been defined on entertainment types, rather than on entertainments. This choice is justified by the intention to keep the model simple while increasing its expressiveness. Indeed, the transition preference between entertainment types can be computed in two ways: (i) directly, by using the knowledge provided by domain experts, or (ii) indirectly, by using the previously collected logs ([D4]) and counting the number of times (frequency) in which the two genres appear close to each other in a historical sequence. Conversely, defining the transition preference between single entertainments would require additional information that cannot be statistically derived from past user behaviours, but have to be manually collected for each individual user and can be difficulty generalized to new items never considered before.

Proposition 9 (From logs to entertainment transition preference)**.** *Let us consider a set of records S containing tuples of the form $\langle u, e_1, \ldots, e_n \rangle$ storing that the user u has performed the sequence of activities $\{e_1, \ldots, e_n\}$. The transition preference between two entertainment types $e_i.\mathsf{gen}$ and $e_j.\mathsf{gen}$ can be obtained as:*

$$\sigma(e_i.\mathsf{gen}, e_j.\mathsf{gen}) = \frac{|\{s \in S \mid \exists h(e_h.\mathsf{gen} = e_i.\mathsf{gen} \wedge e_{h+1}.\mathsf{gen} = e_j.\mathsf{gen})\}|}{|S|}$$

The computation of the transition preference between two entertainment types is performed by taking into consideration the "one-step transition probability", which is the probability of transition from one state to another in a single step. It is also known as Homogeneous Markov Chain, since the transition probability from one state to another is independent from the time index n. In other words, the transition preference is computed by taking into consideration only the two genres at hand (i.e., $e_i.\mathsf{gen}$ and $e_j.\mathsf{gen}$) without considering the entire composition of the sequence (i.e., the entertainments that precede e_i). This can be considered a simplification, but the construction of a more complex model could be very difficult to achieve in real situations, due to both the number of possible combinations to be considered and the corresponding number of data instances to be acquired. Moreover, such additional cost not necessarily produces a more accurate recommendation model. Therefore, Eq. 7 may be considered a good compromise between the level of details and the computational cost.

Given this notion of transition preference for entertainment types, the following proposition computes the dynamic user preference for a given entertainment in a partial sequence.

Definition 10 (Sequential user preference)**.** *Given a user u in the context $c(g_t)$, a partial sequence of entertainment suggestions $\varepsilon = \langle e_1, \ldots, e_k \rangle$ and a new entertainment e, the user preference $\bar{p}(u, c(g_t), \varepsilon, e)$ is defined in the following way:*

$$\bar{p}(u, c(g_t), \varepsilon, e) = \bar{p}(u, c(g_t), e) \cdot \sigma(e_k.\mathsf{gen}, e.\mathsf{gen}) \tag{7}$$

where the function $\bar{p}(u, c(g_t), e)$ has been defined in Prop. 7 and returns the preference of the user u for an entertainment e in the context $c(g_t)$, while σ is the similarity function defined in Def. 8.

Given the individual contextual preferences, the following section discusses how we can derive group contextual preferences.

3.2.3. From Individual Contextual Preferences to Group Contextual Preferences

Some different strategies have been described in the literature in order to determine the preference of a group starting from the preferences of the individual members [7]. These techniques essentially differ for two main aspects: (i) the emphasis placed on the individual satisfaction compared to the satisfaction of the majority of the group, (2) the use of only the relative position of items in each individual's preference list, or of also the strengths of these preferences. In the same work the authors experimentally conclude that humans mainly use as strategies the "Average Without Misery Strategy" and "Least Misery Strategy". In the first case, we make a list of ratings with the average of the individual ratings, but without considering the items with a score below a certain threshold for some users (i.e., misery). The idea is to find the optimal decision for the group without making some group members really unhappy with this decision. In the second case, we assume that the group is happy as its least happy member: however, in this case we have the disadvantage that the minority opinion could dictate the overall group. In both cases, what we can conclude is that when a group satisfaction is considered, preference balancing is more important than a mere global preference maximization, namely a group is happy if everybody is equally happy or miserable. However, the experiments also highlight that despite the importance of balancing, when a sequence of recommendations is provided, each member expects to be satisfied soon or later in the sequence.

Given such considerations, in this paper, we consider two distinct measures, i.e. *group satisfaction* and *group preference balancing*, as two distinct functions that have to be optimized in order to achieve both goals: properly satisfying each single group members without the detriment of someone.

Definition 11 (Group preference or satisfaction)**.** *Let $c(g_{t_i}) = \langle \rho(g_{t_i}), \tau(t_i) \rangle$ be a dynamic context for a group g_{t_i} at temporal instant t_i, $\varepsilon = \langle e_1, \ldots, e_n \rangle$ a partial experience built so far, and e_j be an entertainment. The overall preference of the group is defined as a weighted average of the user preferences where the weight assigned to each member depends on the most likely evolution of the group.*

$$\bar{p}(g_{t_i}, c(g_{t_i}), \varepsilon, e_j) = \sum_{u_k \in g_{t_i}} w(u_k, g_{t_i}, g_{t_{i+1}}) \cdot \bar{p}(u_k, c(g_{t_i}), \varepsilon, e_j) \tag{8}$$

The weight $w(u_k, g_{t_i}, g_{t_{i+1}})$ assigned to a user u_k depends on the possible evolution of the group in a way that if a member is more likely to leave the group in the near future, she will be satisfied early in the sequence, as formalized by the following proposition.

Proposition 12. *Let $c(g_{t_i}) = \langle \rho(g_{t_i}), \tau(t_i) \rangle$ be a dynamic context for a group g_{t_i} at time instant t_i, such that $g_{t_i} \cap g_{t_{i+1}} \neq \emptyset$, and let ε be a partial experience* *and* e_j *be an entertainment. The weight of the users in* $g_{t_i} \setminus g_{t_{i+1}}$ *is greater than the weight of the users in* $g_{t_i} \cap g_{t_{i+1}}$*, during the computation of* $\bar{p}(g_{t_i}, c(g_{t_i}), \varepsilon, e_j)$.

The rationale behind the above proposition is that the users who will leave the group have no chance to be satisfied in the future, while the users who remain in the group can be satisfied also by the next suggestions. Notice that two distinct concepts of time spent by a group together can be recognized: (a) the amount of time the group members decide to spend together (i.e., *available time*), and (b) the amount of time the group members effectively spent together. While the first one could be considered a constraint of the optimization problem and after this period of time the group necessarily evolves into a different one, the second one reflects the fact that if some members particularly dislike the activities performed so far, they can decide to prematurely leave the group. The final aim of a good system for producing sequences of recommendations for dynamic groups is to ensure that the second interval will be equal to the first one. When we talk about possible group evolutions, we want to investigate the possible transformations of the group at the end of the available time. Indeed, after this defined period, a subset of the original group g_{t_i} could decide to perform activities together and this new group $g_{t_{i+1}}$ represents the evolution of g_{t_i}. Let us consider for instance a group with type $\{\mathsf{adult}, \mathsf{kid}\}$ and an available time of an hour to spend together watching TV during the evening. After that period of time, the group will probably evolve into another one with type $\{\mathsf{adult}\}$, since the kids will go to bed before their parents.

The computation of $w(u_k, g_{t_i}, g_{t_{i+1}})$ can be performed in the following way.

Proposition 13 (From logs to group evolutions)**.** *Let* $\Delta(g_{t_i}, t_{i+1})$ *the set of possible evolutions of the dynamic group* g_{t_i} *at time instant* t_{i+1}*, and let* $T \subseteq \Delta(g_{t_i}, t_{i+1})$ *the subset of evolutions in which the type of the user* u_k *is still present:*

$$T = \{\rho(g_{t_i}) \in \Delta(g_{t_i}, t_{i+1}) \mid u_k.\mathsf{type} \in \rho(g_{t_{i+1}})\}$$

Given $\Delta(g_{t_i}, t_{i+1})$ *and* T*, the weight of user* u_k *is computed starting from* *the ratio between the cardinalities of these two sets:*

$$w(u_k, g_{t_i}, g_{t_{i+1}}) = 1 - \frac{|T|}{|\Delta(g_{t_i}, t_{i+1})|} \tag{9}$$

From the formula it is clear that the weight decreases as the number of possibile evolutions containing the user type increases, namely as the probability of the user to remain in the group increases. We can also notice that the sum of all weights is equal to 1.

Given the notion of preference or satisfaction of a dynamic group for a recommendation sequence, the last main aspect to consider is preference balancing. We need to define one or more functions that relate the satisfaction of each group member to the satisfaction of the other ones. For this purpose, we define the concept of group preference balancing as the combination of two functions: the *min-max ratio* and the *Jain's metrics* [19]. We have chosen to consider both metrics because the first one gives emphasis to the gap between the last and the highest preference, while the second one ensures that closer preferences are achieved by all members. Both metrics have been adapted in order to consider (a) dynamic preferences which evolve with contextual features, and (b) sequences of recommendations instead of single ones.

Definition 14 (Group preference balancing)**.** *Let $c(g_{t_i}) = \langle \rho(g_{t_i}), \tau(t_i) \rangle$ be a dynamic context for a group g_{t_i} at temporal instant t_i, $\varepsilon = \langle e_1, \dots, e_n \rangle$ a partial experience built so far, and e_j be an entertainment. The overall balancing of the group preferences is defined through the following two functions:*

$$r_m(g_t, c(g_t), \varepsilon, e) = \frac{\min\{\bar{p}(u, c(g_t), \varepsilon, e), \forall u \in g_t\}}{\max\{\bar{p}(u, c(g_t), \varepsilon, e), \forall u \in g_t\}} \tag{10}$$

$$r_j(g_t, c(g_t), \varepsilon, e) = \frac{\left(\sum_{u \in g_t} \bar{p}(u, c(g_t), \varepsilon, e)\right)^2}{|g_t| \cdot \sum_{u \in g_t} \bar{p}(u, c(g_t), \varepsilon, e)^2} \tag{11}$$

Given all the ingredients of our recommendation system, we can define a recommendation query as follows.

Definition 15 (Recommendation query)**.** *A group of users looking for a recommendation submits a query Q to the system containing the following information:*

- *the group composition* $g_t = \{u_1, \ldots, u_n\}$*;*
- *the initial context* $c(g_{t_0}) = \langle \rho(g_{t_0}), \tau(t_0) \rangle$*: it includes the group type and the temporal characterization of* t_0*;*
- *the desired duration as an interval* $(d_{\min}, d_{\max})$*;*
- *the mandatory maximum duration* $TD_{\max}$*;*
- *the mandatory maximum available budget* $b_{\max}$*;*

Notice that, while the start time t_0, the maximum duration $\mathrm{TD}_{\max}$ and the maximum budget $b_{\max}$ are considered as mandatory constraints, the desired duration is intended as a desiderata: the recommended experiences should have an overall duration close to the desired one. We recall that while the group composition is used only to identify the group members and to retrieve their individual contextual preferences, the context is defined by the group type and the temporal characterization.

Among all possible sequences of entertainments that satisfy the given constraints, the exploration of the search space is guided by the value of the objective functions.

Definition 16 (Objective functions)**.** *Given a recommendation query* Q*, an experience* $\varepsilon = \langle e_1, \ldots e_n \rangle$ *and a dynamic context* $c(g_t) = \langle \rho(g_t), \tau(t) \rangle$*, the considered objective functions to be minimized are:*

- $f_d(\varepsilon, c(g_t)) = \begin{cases} w_a \cdot (d_{\min} - \delta(\varepsilon)) & \textit{if } \delta(\varepsilon) < d_{\min} \\ w_b \cdot (\delta(\varepsilon) - d_{\max}) & \textit{if } \delta(\varepsilon) > d_{\max} \\ d_{\max} - \delta(\varepsilon) & \textit{otherwise} \end{cases}$

  f_d *computes the difference between the actual duration of the experience* ε *and the desired duration, i.e. it is essentially a measure of the empty slots. The two weights* w_a *and* w_b *can be used to consider less appealing a duration smaller than the minimum desired one, w.r.t. a duration bigger than the maximum desired one.*

- $f_e(\varepsilon, c(g_t)) = \sum_{e_j \in \varepsilon} \mid e_j.\mathsf{dur} - (e_j.\mathsf{end} - e_j.\mathsf{start}) \mid$
  f_e computes the sum of the portions of entertainments the group did not enjoy during the sequence, which has to be minimized.

- $f_h(\varepsilon, c(g_t)) = |\bigcup_{u_k \in g_t} H(u_k, t) \cap \bigcup_{e_j \in \varepsilon} e_j|$
  f_h counts the number of entertainments already viewed by each member of the group, considering its past viewing. $H(u_k, t)$ is the past history of the user u_k, i.e. it is the set of entertainments enjoyed by u_k in the past. The function f_h depends on the time instant t (i.e., the context) in which it is computed.

- $f_s(\varepsilon, c(g_t), t) = n - \sum_{i=1}^{n} \bar{p}(g_t, c(g_t), \varepsilon, e_j)$
  The function f_s minimizes the loss of preference, in the formula n is the number of entertainments in the final global experience ε and 1 is the possible maximum degree of satisfaction (preference) for any entertainment in ε, while $\bar{p}(g_{t_i}, c(g_{t_i}), \varepsilon, e_j)$ has been defined in Def. 11.

- $f_m(\varepsilon, c(g_t), t) = n - \sum_{i=1}^{n} r_m(g_t, c(g_t), \varepsilon, e_i)$
  The function f_m minimizes the gap between the minimum and maximum satisfaction of each group member w.r.t. the overall sequence. For each entertainment e_i in the experience ε, the value $r_m(g_t, c(g_t), \varepsilon, e_i)$ is less than or equal to one. In particular, it is equal to one when the gap is minimum, so in the best case the summation for all entertainments is equal to n.

- $f_j(\varepsilon, c(g_t)) = \dfrac{n}{|g_t|} - \sum_{i=1}^{n} r_j(g_t, c(g_t), \varepsilon, e_i)$
  The function f_j minimizes the difference between the preferences of the group members w.r.t. the overall sequence. For each entertainment e_i in the experience ε, the value $r_j(g_t, c(g_t), \varepsilon, e_i)$ is at most equal to $1/|g_t|$ when the preferences of the group members for the entertainment e_i are closer to each one. Therefore, given a sequence of n entertainments, the best value is achieved when the value of the summation becomes equal to n.

Given the functions defined above, the group recommendation problem can be formulated as an optimization problem:

$$\begin{aligned} \underset{\varepsilon}{\text{Minimize}} \quad & \langle f_d, f_e, f_n, f_s, f_m, f_j \rangle \\ \text{subject to} \quad & \delta(\varepsilon) < \mathrm{TD}_{\max} \\ & \gamma(\varepsilon) < b_{\max} \end{aligned} \tag{12}$$

The six objective functions can be combined in a single function $\bar{f} : \mathcal{E} \rightarrow \mathbb{R}^6$ which, given an experience $\varepsilon \in \mathcal{E}$, returns a tuple as value. It follows that it is possible to establish only a partial order between solutions. In particular, a *dominance* relation is defined between two solutions, represented as $s \prec s'$. We say that s dominates s' (denoted as $s \prec s'$), if and only if s is better than s' in at least one of the objective functions and equivalent in the other ones.

Notice that the proposed solution is general enough to be easily adapted to other application domains. In particular, the optimization functions described above can be enriched or substituted with other functions that better capture the essence of the recommendation problem.

4. Proposed Solution: a MOSA Approach

In the previous section we have formalized the problem of producing sequences of recommendations for dynamic groups as an optimization problem. The identification of an exact solution for such kind of problem is known to be computationally hard. Indeed, it is a combinatorial optimization problem whose complexity increases as the number of objective functions increases [20]. Moreover, while single objective optimization problems are intractable time-wise but have finite space (memory) requirements, multi-objective optimization problems, on the other hand, are intractable both with regards to time and space [21].

Therefore, several different heuristics have been proposed in the literature in order to efficiently produce a solution in a reasonable amount of time: from greedy algorithms to simulated annealing techniques. In this paper we choose to apply the Multi-Objective Simulated Annealing (MOSA) [22] for two main reasons: (1) it is able to reach a global optimum if annealed sufficiently slowly [23], while other solutions, like the greedy algorithm, can stuck in local optima; (2) in the search space exploration, instead of using a random artificial solution as starting point, we can rely on the available historical data, i.e., the past experiences of similar groups in the same context. This second aspect of the MOSA technique gives an important role to the data collected about similar groups in similar contexts, which is also known in the recommendation system field as collaborative filtering.

Before describing the MOSA solution, we show how the pre-processing steps depicted in Fig. 1 can be efficiently implemented with a MapReduce approach.

Algorithm 1: Mapper phase for the computation of [D1][D2].

class *D1D2Mapper*
    method *setup*()
        $R_{map} \longleftarrow \emptyset; R^*_{map} \longleftarrow \emptyset$
    method *map*($\langle \_, \langle u, g_t, start, end, e \rangle \rangle$)
        $c(g_t) \longleftarrow \langle \rho(g_t), \tau(start) \rangle$
        $v \longleftarrow R_{map}.\mathtt{get}(\langle u, c(g_t), e.\mathsf{gen} \rangle) + (end - start)$
        $R_{map}.\mathtt{put}(\langle u, c(g_t), e.\mathsf{gen} \rangle, v)$
        $t \longleftarrow R^*_{map}.\mathtt{get}(\langle u, c(g_t) \rangle) + (end - start)$
        $R^*_{map}.\mathtt{put}(\langle u, c(g_t) \rangle, t)$
    method *cleanup*()
        foreach $(k, v) \in R_{map}$ **do**
            $k' \longleftarrow \langle k.u, k.c(g_t) \rangle$
            $\mathtt{write}(k, \langle v, R^*_{map}.\mathtt{get}(k') \rangle)$
        end

4.1. Pre-Processing Steps

The MOSA algorithm proposed in this paper requires that some historical data be analysed through a set of pre-processing steps, as illustrated in Fig. 1. These inputs regard individual dynamic preferences (i.e., [D1],[D2]), the possible group evolutions (i.e., [D3]) and the preferred transitions between genres in a sequence (i.e., [D4]).

The computation of the individual dynamic preferences from past logs about users' activities is described in Prop. 7 and 10. This computation can be effectively implemented with a MapReduce job as illustrated in Alg. 1-2.

Algorithm 2: Reducer phase for the computation of [D1][D2].

class *D1D2Reducer*
    method *reduce*$(\langle k, \{(v_1, t_1), \dots (v_n, t_n)\}\rangle)$
        $v \longleftarrow 0; t \longleftarrow 0$
        for $i = 1 \dots n$ **do**
            $v \longleftarrow v + v_i; t \longleftarrow t + t_i$
        end
        `write`$\left(k, \dfrac{v}{t \cdot n}\right)$

Let us consider a log dataset $D_{1,2}$ containing records of type $\langle u, g_t, start, end, e\rangle$ representing the fact that user u enjoys the entertainment e when she was in a group g_t for a period between *start* and *end*. This dataset can be passed as input of the MapReduce job, which is responsible for computing the myopic user preferences described in Prop. 7 and will be used at runtime during the construction of a sequence of suggestions, in particular for the computation of the dynamic preferences (see Prop. 10). During the map phase in Alg. 1, each mapper processes a record at time and builds two auxiliary data structures called R_{map} and R^*_{map}. They are both associative arrays: R_{map} stores for each user u, context $c(g_t)$ and entertainment type e.gen, the global duration of the visions performed by u in the context $c(g_t)$ relatively to entertainments with type e.gen; while R^*_{map} stores a similar value but without distinguishing between different

Algorithm 3: Mapper phase for the computation of [D4].

```
class D4Mapper
    method setup()
        E_map ⟵ ∅; T_map ⟵ ∅
    method map(ε = ⟨e_1, ..., e_n⟩)
        for i = 1 ... n − 1 do
            if e_i.gen ≠ e_j.gen then
                v ⟵ E_map.get(⟨e_i.gen, e_{i+1}.gen⟩)
                E_map.put(⟨e_i.gen, e_{i+1}.gen⟩, v + 1)
                p ⟵ T_map.get(e_i.gen)
                T_map.put(e_i.gen, p + 1)
            end
        end
    method cleanup()
        foreach (⟨e_i, e_j⟩, v) ∈ E_map do
            write(⟨e_i, e_j⟩, v, T_map.get(e_i))
        end
```

entertainment types. These two associative arrays are used by the reducers in order to compute the value in Eq. 6. Notice that regarding the map phase, the key associated to each mapper input is negligible, so it has been represented by the symbol "_". Conversely, each reducer input is represented by the list of partial values associated to each user, context and entertainment types that have been computed by the various mappers. The goal of the reducer is to combine such partial results in order to obtain the final value in Eq. 6. More than one reducer can be instantiated: one for each user, context and entertainment type.

In order to compute the dynamic user preferences formalized in Eq. 7 it is necessary to determine the transition probability between two entertainments types e_i.gen and e_j.gen. This corresponds to the pre-processing step [D4] described in Prop. 9 which can be implemented with the MapReduce job in Alg. 3-

Algorithm 4: Reducer phase for the computation of [D4].

class *D4Reducer*
  method *reduce*$(\langle k, \{(v_1, t_1), \ldots, (v_n, t_n)\}\rangle)$
    $v \longleftarrow 0;\ t \longleftarrow 0$
    for $i = 1 \ldots n$ **do**
      $v \longleftarrow v + v_i;\ t \longleftarrow t + t_i$
    end
    `write`$\left(k, \dfrac{v}{t}\right)$

4. Let us consider a dataset D_4 containing past sequences of entertainments $\langle e_1, \ldots, e_n\rangle$: it can be built starting from the dataset $D_{1,2}$ by concatenating consecutive entertainments enjoyed by the same group of users. Two entertainments e_i and e_j are considered consecutive if $\mid e_j.end - e_i.start \mid < th$, where th is a threshold appropriately determined on the basis of the considered application domain. The mapper in Alg. 3 simply maintains a counter for each pair of genres e_i.gen and e_j.gen which appear next to each other in a sequence (i.e. through the associative array E_{map}), as well as a counter of the number of transactions collected in the logs which starts from a given e_i.gen (i.e., in the associative array T_{map}). The partial counters produced by the mappers are finally combined by the reducers: notice that more than one reducer can be instantiated in parallel. The reducers in Alg. 4 compute, for each pair of genres, (e_i.gen, e_j.gen), the total amount of transitions involving them (i.e., variable v) and then divide such value for the total number of transitions starting from the source genre e_i.gen (i.e., variable t).

The final pre-processing step to be performed is the one that determines, for each possible pair of group types, the probability of an evolution from the source group type to the target group type ([D3]). The most important part regards the correct identification of an evolution given a log dataset as the $D_{1,2}$ previously described. The details are reported in Alg. 5-6. During the map phase, the records are grouped w.r.t. to the user u, so that the reducer receives for each user u the corresponding list of records. In each reducer in Alg. 6, these records are initially sorted by the *start* timestamp (i.e., the timestamp at which the given entertainment has been started). Two records are considered consecutive if and only if the difference between the start of the second and the end of the first is less than a given threshold th. In case of two consecutive records, if the corresponding group types are different, then an interesting transition is registered in an associative array G. The key of this map is the pair of types $\langle \rho(v_i.g_t), \rho(v_j.g_t) \rangle$, while the value is a counter of the frequency of a transition from a group type $\rho(v_i.g_t)$ to another type $\rho(v_j.g_t)$. A similar array T is maintained: it counts the number of encountered transitions starting from a given group type and is used to produce a normalized value (see the value of the denominator in Eq. 9).

The results of these pre-processing steps are used during the online annealing procedure in order to build the best sequences of recommendations for groups of users, given a recommendation query Q and the available set of entertainments $\mathcal{O}$. As discussed in the previous sections, in order to provide an efficient online MOSA solution, a MapReduce implementation is presented in the following section.

Algorithm 5: Mapper phase for the computation of [D3].

class *D3Mapper*
    method *setup*()
        $M \longleftarrow \emptyset$
    method *map*($\langle \_, \langle u, g_t, start, end, e \rangle \rangle$)
        $l \longleftarrow M.\texttt{get}(u) \cup \{\langle u, g_t, start, end, e \rangle\}$
        $M.\texttt{put}(u, l)$
    method *cleanup*()
        foreach $\langle k, v \rangle \in M$ **do**
            $\texttt{write}(k, v)$
        end

Algorithm 6: Reducer phase for the computation of [D3].

class *D3Reducer*

    method *setup*()

        $G \longleftarrow \emptyset$; $T \longleftarrow \emptyset$

    method *reduce*($\langle k, l = \{(v_1, \ldots, v_n\}\rangle$)

        $l.\texttt{sortBy}(start)$

        for $i = 1 \ldots n-1$ **do**

            if $(v_{i+1}.start - v_i.end) < th \wedge \rho(v_i.g_t) \neq \rho(v_{i+1}.g_t)$ **then**

                $c \longleftarrow G.\texttt{get}(\langle \rho(v_i.g_t), \rho(v_{i+1}.g_t)\rangle)$

                $G.\texttt{put}(\langle \rho(v_i.g_t), \rho(v_{i+1}.g_t)\rangle, c+1)$

                $t \longleftarrow T.\texttt{get}(\rho(v_i.g_t))$

                $T.\texttt{put}(\rho(v_i.g_t), v+1)$

            end

        end

    method *cleanup*()

        foreach $\langle\langle \rho(v_i.g_t), \rho(v_j.g_t)\rangle, v\rangle \in G$ **do**

            $\texttt{write}\left(\langle \rho(v_i.g_t), \rho(v_j.g_t)\rangle, \dfrac{v}{T.\texttt{get}(\rho(v_i.g_t))}\right)$

        end

MapReduce Multi-Objective Simulated Annealing

The main characteristic of the MOSA technique is that the exploration of the search space does not start from a random point, but from an existing solution s which is sightly modified in some way (i.e., perturbed) during the exploration by applying atomic transformations. Besides to the good theoretical properties of this choice, it also seems to be the most appropriate solution for the considered problem because it is more reasonable to obtain good suggestions by starting from past historical sequences of activities, since they capture the real behaviour of a similar group in a similar context, instead of randomly choosing a set of activities from scratch (in accordance with the idea of collaborative filtering).

From a theoretical point of view, at each step of the MOSA technique, given a current solution s_1, an elementary change (i.e. *perturbation*) is applied to s_1 obtaining a new solution s_2. The new solution s_2 is chosen in place of the previous one, called s_1, with a probability that depends on both the value of the objective functions and a global temperature parameter, which is progressively decreased during the execution, resembling what happens in physical annealing procedures. This temperature parameter is what allows the technique to reach a global optimum, instead of stacking in a local one. Indeed, at high temperatures the probability to choose a solution which is worsen than the current one is higher, thus to provide a wider exploration of the search space, while the temperature decreases, this probability is reduced and only better solutions are considered in place of the current one in the final steps.

Concerning the possible perturbations applied to a current solution (i.e, a sequence of activities), in the considered scenario we apply the following elementary changes: (i) the *removal* or *addition* of a single entertainment, (ii) the *replacement* of an entertainment with a new one, or (iii) a change in the *order* of the entertainments.

Since the MOSA technique considers multiple objective functions to be optimized, it does not provide a unique optimal solution, but a set of equally good solutions, called *Pareto-set*, which is built around the concept of dominance.

Definition 17 (Dominance)**.** *An experience ε_i dominates another experience ε_k, denoted as $\varepsilon_i \prec \varepsilon_k$, if it is better in at least one objective function and equivalent in all the remaining ones:*

$$\begin{aligned}
&\varepsilon_i \prec \varepsilon_k \iff \\
&\qquad \forall f \in \langle f_d, f_e, f_n, f_s, f_m, f_j \rangle\ f(\varepsilon_i, c(g_t)) \leq f(\varepsilon_k, c(g_t)) \land \\
&\qquad \exists f \in \langle f_d, f_e, f_n, f_s, f_m, f_j \rangle\ f(\varepsilon_i, c(g_t)) < f(\varepsilon_k, c(g_t))
\end{aligned}$$

Given the notion of dominance, we can introduce the concept of Pareto-set.

Definition 18 (Pareto-set)**.** *Given a set S of experiences, the Pareto-set $\mathcal{PS} \subseteq S$ is the set of mutually non-dominating experiences. Two experiences ε_i and*

Algorithm 7: Job for the initialization of the initial Pareto-set $\mathcal{PS}_{\text{init}}$.

```
job PsInit(Q)
    jobConf ⟵ {Q}
    {PS_1, ..., PS_n} ⟵ PsInitMapper
    PS_init ⟵ PsInitReducer
```

ε_j are mutually non-dominating if and only if neither of them dominates the other.

A solution s not dominated by any other solution in S is called *Pareto-optimum*. Given a Pareto-set $\mathcal{PS}$, we can compute the corresponding *Pareto-front* $\mathcal{F} \subseteq \mathbb{R}^6$, namely the set of points in the objective space corresponding to the solutions in the $\mathcal{PS}$.

Definition 19 (Pareto-front)**.** *Given a Pareto-set $\mathcal{PS}$, the corresponding Pareto-front $\mathcal{F} \subseteq \mathbb{R}^6$ is the image of $\mathcal{PS}$ in the objective space, namely the set of points:*

$$\mathcal{F} = \{\bar{f}(\varepsilon) \mid \varepsilon \in \mathcal{PS}\}$$

where $\bar{f} : \mathcal{E} \to \mathbb{R}^6$ has been introduced at the end of Sect. 3.2 as the combination of the six objective functions; given an experience $\varepsilon \in \mathcal{E}$, it returns as value a tuple representing a point in the objective space.

Given all these definitions, we can say that the overall goal of the MOSA algorithm is to move the current Pareto-front, computed from the current Pareto-set, towards the optimal Pareto-front (i.e., the Pareto-front of the actual Pareto-optimum solutions) while encouraging the diversification of the candidate solutions.

As mentioned at the beginning of this section, given two experiences $\varepsilon_{\text{curr}}$ and ε_{new}, the probability to make a transition from the current solution $\varepsilon_{\text{curr}}$ towards the new one ε_{new}, depends upon a global *temperature* parameter $\mathcal{C}$ and a measure of goodness of the two solutions, called *energy*.

Definition 20 (Energy). *The energy of a solution ε, denoted as $E(\varepsilon, \mathcal{F})$, measures the portion (number of solutions) of the current Pareto-front that dominates ε, i.e., $E(\varepsilon, \mathcal{F}) = |\{v \in \mathcal{F} \mid v \prec \bar{f}(\varepsilon)\}|$.*

Notice that the energy of an experience ε belonging to the Pareto-front is 0.

Algorithm 8: Job for the initialization of the initial Pareto-set $\mathcal{PS}_{\text{init}}$.

class *PsInitMapper*
    method *setup*()
        $\mathcal{PS}_{\text{map}} \longleftarrow \emptyset$; $Q \longleftarrow$ retrieve from job configuration
    method *map*($\_, \varepsilon$)
        if ε *satisfies* Q **then**
            `updateParetoSet`($\mathcal{PS}_{\text{map}}, \varepsilon$)
        end
    method *cleanup*()
        `write`($\_, \mathcal{PS}_{\text{map}}$)
class *PsInitReducer*
    method *reduce*($\_, \{\mathcal{PS}_1, \mathcal{PS}_2, \dots\}$)
        $\mathcal{PS}_{\text{init}} \longleftarrow \emptyset$
        for $\mathcal{PS}_i \in \{\mathcal{PS}_1, \mathcal{PS}_2, \dots\}$ **do**
            for $\varepsilon \in \mathcal{PS}_i$ **do**
                `updateParetoSet`($\mathcal{PS}_{\text{init}}, \varepsilon$)
            end
        end
        `write`($\mathcal{PS}_{\text{init}}$)

The application of the MOSA technique to the problem of producing sequences of recommendations for groups produces a set of equally good suggestions, which can be proposed to the group as possible alternatives. This can lead to two interesting extensions that we plan as future work: (i) the use of explanation techniques for justifying the suggestions to the users and promoting a conscious choice, these explanations can be automatically (or semi-automatically) produced on the basis of the values of the objective functions; (ii) a feedback about the actual choices performed by a group among the possible alternatives, can be used to better calibrate future suggestions.

Alg. 7-11 illustrate a MapReduce implementation of the MOSA technique, this implementation parallelises the computation and reduces the overall time required to produce the suggestions. This is an important aspect to be achieved, since this task has to be performed on-line as soon as new queries are submitted to the system.

A first MapReduce job is used to extract the initial Pareto-set $\mathcal{PS}_{\text{init}}$ starting from the past historical sequences and the current query Q (see Alg. 7). For each historical sequence ε in the log data that satisfies the query Q, the $\mathcal{PS}_{\text{init}}$ set is updated through the function `updateParetoSet`(), which checks if the current solution ε can be included in the set and eventually removes all the solutions in $\mathcal{PS}_{\text{init}}$ that are dominated by it. The construction of $\mathcal{PS}_{\text{init}}$ can be performed in parallel through the map and reduce procedures in Alg. 8: each mapper produces a partial Pareto-set $\mathcal{PS}_i$ starting from the processed experiences, then the reducer can combine them by removing the dominated solutions and returning $\mathcal{PS}_{\text{init}}$. The presence of a single reducer is not a limitation, since the amount of work it has to do is very limited.

Algorithm 9: MapReduce job implementing MOSA.

job *Mosa*(Q)
    $\mathcal{PS} \longleftarrow$ `PsInit`(Q)
    $t \longleftarrow t_{\text{init}}$
    while $t > t_{min}$ **do**
        $jobConf \longleftarrow \{\mathcal{PS}, Q, t\}$
        $\{\mathcal{PS}_1, \dots \mathcal{PS}_n\} \longleftarrow$ `MosaMapper`
        $\mathcal{PS} \longleftarrow$ `MosaReducer`
        `updateTemperature`(t)
    end

Algorithm 10: MapReduce job implementing MOSA.

```
class MosaMapper
    method setup()
        PS, Q, t ⟵ retrieve from job configuration
        S_i ⟵ ∅
    method map(_, ε)
        S_i ⟵ S_i ∪ {ε}
    method cleanup()
        foreach ε ∈ S_i do
            PS_i ⟵ Annealing(PS, ε, Q, t)
            write(_, PS_i)
        end
class MosaReducer
    method reduce(_, ⟨PS_1, PS_2, ...⟩)
        PS ⟵ ∅
        for PS_i ∈ ⟨PS_1, PS_2, ...⟩ do
            for ε ∈ PS_i do
                updateParetoSet(PS, ε)
            end
        end
        return (_, PS)
```

Algorithm 11: Annealing procedure.

function *Annealing*$(\mathcal{PS}, \varepsilon, Q)$

  $\mathcal{F} \longleftarrow$ `computeParetoFront`$(\mathcal{PS})$

  for $i = 1 \ldots maxPert$ **do**

    $\varepsilon' \longleftarrow$ `perturb`$(\varepsilon,\, t)$

    $\mathcal{F}' \longleftarrow \mathcal{F} \cup \bar{f}(\varepsilon')$

    $\Delta_E \longleftarrow$ `computeEnergyDiff`$(\varepsilon', \varepsilon, \mathcal{F}', Q)$

    $P \longleftarrow \min(1, \exp(-\Delta_E / t))$

    if $rand(0,1) < P$ **then**

      `removeDominated`$(\mathcal{PS}, \varepsilon', \mathcal{F}, \bar{f}(\varepsilon'), Q)$

      $\mathcal{PS} \longleftarrow \mathcal{PS} \cup \{\varepsilon'\};\ \mathcal{F} \longleftarrow \mathcal{F} \cup \{\bar{f}(\varepsilon')\}$

      $\varepsilon \longleftarrow \varepsilon'$

    end

  end

  return $\mathcal{PS}$

Proposition 21 (`PsInit` correctness)**.** *The Pareto set $\mathcal{PS}_{init}$ produced by the job in Alg. 7 is the same as the one produced by a sequential algorithm.*

Proof. A sequential computation of the Pareto set produces a collection of solutions $\mathcal{PS}_{\text{init}}$ such that $\forall s_1, s_2 \in \mathcal{PS}_{\text{init}} (s_1 \not\prec s_2)$ nor $(s_1 \not\prec s_2)$. Clearly, each mapper in Alg. 8 is able to produce a set $\mathcal{PS}_i$ with such characteristic, but when we consider the partial results individually produced by the mappers there can exist a pair of solutions $s_h \in \mathcal{PS}_i$ and $s_k \in \mathcal{PS}_j$ such that $s_h \prec s_k \vee s_k \prec s_h$. Anyway, all these situations are solved by the single reducer which removes any solution in a partial Pareto set that is dominated by another solution in another partial Pareto set. □

The initial Pareto-set $\mathcal{PS}_{\text{init}}$ produced by this preliminary job represents the input for the following job, which actually implements the MOSA technique. Notice that for very frequent queries the corresponding $\mathcal{PS}_{\text{init}}$ can be pre-computed, stored and updated off-line, in order to improve the overall performances of the recommendation system.

The MapReduce job in Alg. 9 represents the core of the recommendation system. Given a recommendation query Q including the initial context $c(g_{t_0})$, the desired duration $(d_{\min}, d_{\max})$, the mandatory maximum duration $TD_{\max}$ and the mandatory maximum budget $b_{\max}$, the job computes a set of suggestions for the group. More specifically, after the initial Pareto-set $\mathcal{PS}$ has been computed (line 2), for each temperature value t, the job executes the mappers and the reducers tasks in Alg. 10. Each mapper performs the annealing operation in Alg. 11 on a subset of the global initial solution set S. Function `Annealing` essentially applies a perturbation to the current solution ε obtaining a new solution ε', then it evaluates the probability of taking ε' in place of the current one. This probability (see line 7) depends on the difference between the energy of the two solutions (see line 6) and the temperature parameter t. Then if the new solution is chosen in place of the current one, the Pareto-set $\mathcal{PS}$ and the Pareto-front $\mathcal{F}$ are accordingly updated. This operation is performed a predefined number of times (i.e., variable *maxPert*) for each temperature value.

The algorithm requires to compute the dominance relation between solutions in two situations: during the computation of the energy difference (see line 6) and during the update the Pareto-set (see line 9). Relating to the dominance, one of the most interesting part is the computation of the function $f_s(\varepsilon, c(g_t), t)$ which requires to estimate the preference of the group for the experience ε in the context $c(g_t)$. The computation of such objective function is illustrated in Alg. 12-13. As you can notice, function `DynUserPreferences` takes care of all the information computed during the pre-processing steps, which have been denoted as $\mathtt{D}_{1,2}$, $\mathtt{D}_3$ and $\mathtt{D}_4$, and produces an array with the individual preferences of all group members (see Eq. 7). First of all, by using the output produced by Alg. 1-2, the preference of each group member w.r.t. any entertainment e is evaluated (line 5); then, if such entertainment is not the first one in the experience, this myopic user preference is corrected by considering the transition preference from the previous entertainment in the sequence ε (line 7), by using

Algorithm 12: Dynamic user preference.

function *DynUserPreferences*(Q, ε)
  $P \longleftarrow \emptyset$
  for $u \in Q.g_t$ **do**
    for $e_i \in \varepsilon$ **do**
      $p_u \longleftarrow \mathtt{D}_{12}.\mathtt{get}(u, Q.c(g_t), e_i.\mathtt{gen})$
      if $i > 1$ **then**
        $p_u \longleftarrow p_u \cdot \mathtt{D}_4.\mathtt{get}(e_{i-1}.\mathtt{gen}, e_i.\mathtt{gen})$
      end
    end
    $P \longleftarrow P \cup \{w \cdot p_u\}$
  end
  return P

the output of Alg. 3-4. The array of individual preferences produced by Alg. 12 is used by function `DynGroupPreference` in Alg. 13, which combines such individual preferences by taking into consideration also the possible group type evolutions computed by Alg. 5-6. First of all, the set of all possible evolutions of a group type $\rho(g_t)$ is retrieved and stored in the set Δ; then, for each group member u_i, we extract a set $T \subseteq \Delta$ containing only the evolutions of $\rho(g_t)$ in which the type of u_i is present (lines 6-12). From the cardinalities of these two sets, the weight w of each user is computed (line 13) and the overall group preference is updated (line 14).

Another interesting part of the dominance is the computation of the group preference balancing, given the dynamic user preferences computed by Alg. 12, as illustrated in Alg. 14. More specifically, given the individual user preferences, they are used to calculate both balancing metrics introduced in Eq. 10-11.

Returning to Alg. 10, after each iteration with a given temperature t, the parallel mappers synchronize themselves through the reducer. The reducer is responsible for combining the distinct Pareto-set computed so far by the various

Algorithm 13: Dynamic group preference.

function *DynGroupPreference*(Q, ε)
  $\{p_1, \ldots, p_n\} \longleftarrow$ `DynUserPreferences`(Q, ε)
  $t_{end} \longleftarrow t + Q.TD_{\max}$
  $\Delta \longleftarrow \mathtt{D}_3.\mathtt{get}(\rho(g_t), \tau(t_{end}))$
  $p \longleftarrow 0$
  for $u_i \in Q.g_t = \{u_1, \ldots, u_n\}$ **do**
    $T \longleftarrow \emptyset$;
    for $X \in \Delta$ **do**
      if $u_i.type \in X$ **then**
        $T \longleftarrow T \cup \{X\}$
      end
    end
    $w \longleftarrow 1 - \frac{|T|}{|\Delta|}$
    $p \longleftarrow p + \{w \cdot p_i\}$
  end
  return p

mappers and for producing an updated global Pareto-set for the next iteration.

Proposition 22 (`Mosa` correctness)**.** *The MapReduce job* `Mosa` *in Alg. 9 provides a good approximation of the optimal solution.*

Proof. With reference to the classification provided in [24], the MapReduce job in Alg. 9 can be classified as a co-operating search of the global optimum. Indeed, given a temperature value t, each mapper performs its annealing by starting from a different solution; then, after a predefined number of perturbations, the partial results produced by the mappers are combined by the reducer. As demonstrated in [24], a co-operative search is the best form of parallel MOSA, since it has the highest probability to find the best solution w.r.t. independent and semi-independent searches. □

Algorithm 14: Preference balancing.

function $\mathit{PreferenceBalancing}(Q, \varepsilon)$

  $\{p_1, \ldots, p_n\} \longleftarrow$ `DynUserPreferences`(Q, ε)

  $r_x \longleftarrow 0; r_n \longleftarrow \infty$

  $j_n \longleftarrow 0; j_d \longleftarrow 0$

  for $i = 1 \ldots n$ **do**

    $r_x \longleftarrow \min(r_x, p_i); r_n \longleftarrow \max(r_n, p_i)$

    $j_n \longleftarrow j + p_i; j_d \longleftarrow j_d + (p_i)^2$

  end

  $r_m = \dfrac{r_x}{r_n}; r_j = \dfrac{(j_n)^2}{n \cdot j_d}$

  return $\{r_m, r_j\}$

We have made available an implementation of the proposed MapReduce technique for Apache Hadoop in our project repository[1] together with the datasets used for the experiments. The following section describes the application of the proposed technique to three real world case scenarios.

5. Evaluation

In this section we evaluate the proposed technique by using three real word datasets: the first one contains data related to past TV watching activities performed by group of users together in different contexts (Auditel dataset), while the last two regard POI visits performed by users and tracked by social media (Foursquare [25, 26] and Gowalla datasets [27, 28]).

5.1. *Auditel dataset: characteristics and analysis*

Dataset characteristics – The Auditel dataset contains approximately 5 million entries regarding the TV viewings performed by almost 8,000 users. We

[1] `https://github.com/smigliorini/emosa`

select about 1.5 million of them which are those longer than three minutes that involve at least two people together. Each record specifies the identifier of both the user and the group, the program that they watched together, as well as the start and the end time. The dataset contains also an electronic program guide (EPG) including the description of about 21,200 distinct programs, as well as their genre. As for the context identification, the *group types* are identified based on the age of the members (i.e., ages are grouped into nine different classes), while the *temporal information* is classified considering eight different time slots and the day of the week. The dataset contains approximately 200 different group types and 60 different temporal characterizations.

Dataset analysis – From a preliminary analysis we can derive the following observations:

1. In the majority of the cases, users watch more than one program in sequence during a day and the average number of programs watched sequentially in a day is about 3.
2. In 80% of the cases, if there is a sequence of program viewings for a user in a day, the number of short viewings in the sequence is greater than the number of long (full) viewings, and in the 46% of such cases the number of short viewings is more than double than the number of full viewings.
3. In 32% of the cases, if there is a sequence of viewings for the same user in a specific day, there are at least two long viewings with some short viewings in the middle.
4. In 80% of the cases, if the user performs more than one viewings in sequence, the longer views are related to the same genre.

From observations 2 and 3, we can conclude that short viewings between two long viewings are essentially the result of a channel surfing activity made in the attempt to find something interesting to watch next. This brings to light the need of providing sequences of recommendations in the TV domain in order to reduce the waste of time during two consecutive activities and improve the value of the overall experience. Conversely, observation 4 confirms that when more than one show is viewed in sequence, the user tends to maintain a certain level of genre uniformity in the viewed programs.

Recommendation example – Fig. 2 illustrates an example of a sequence of viewings performed by a family group together. The initial context is character-

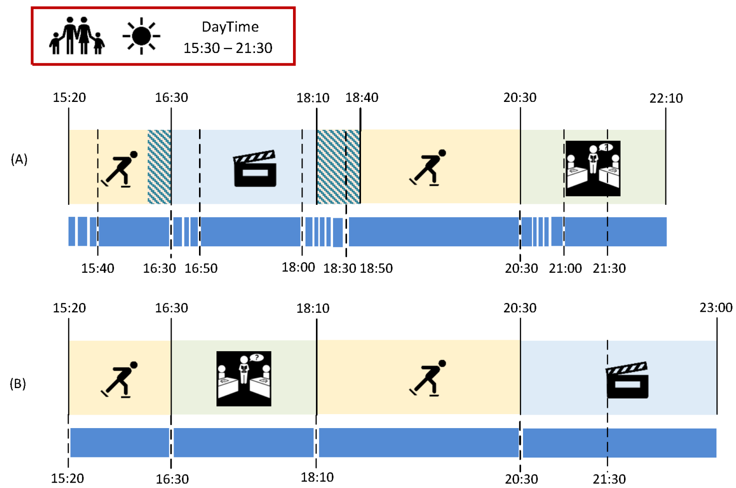

Figure 2: Comparison of two sequences: one contained in the original dataset and one produced by the proposed recommendation technique.

ized by a group of two adults and two children, in a day time period starting at 15:30 and ending at 21:30. Sequence (A) is the one registered in the log file: in particular, the blue segments represent the duration of each viewing performed by the group, while the bigger line with icons describes the scheduling and real durations of each TV show. More specifically, the group starts watching TV together at 15:20 and initially performs some channel surfing before choosing the first program to watch together (see the various small initial blu segments). Only at 15:40 the group finds a sport match they like: the actual beginning of the match is at 15:20, so the group misses the initial part, but the vision continues till the program end at 16:30. Subsequently, the group performs an additional channel surfing before watching a portion of a film and finally finding another sport match to enjoy. As regards this film, what we can observe is that not only the vision starts after its beginning, due to the previous chosen program, but it is also not watched till its end, suggesting that some members do not like it or it is not suitable for them. After prematurely stopping the viewing of the film, the group searches for another show, finally choosing another sport match they start to view at 18:50. Also the view of this match starts after its beginning (at 18.10) but continues until its end. After the end of such match, the group performs some other short viewings until they finally choose a sport commentary program. At 21:30, the group stops to watch program together, perhaps some members go to bed, while some other members continue to watch the show till its end (see the blue line).

What we can observe is that the group really likes to view sport shows together and in particular they like to spent the entire day to watch sport programs (i.e. maintaining a constant mood). Given such information, a possible suggestion is represented by sequence (B) where the two sport matches are the fixed part for two reasons: (i) they are the two shows entirely watched by the group (or at least watched until their end), (ii) their vision cannot be deferred. Between the two sport matches, we place a sport commentary program which seams to be appreciated by the group and maintains a consistent genre between the other two programs. Conversely, the film is placed at the end of the sequence, in this way the users who are still watching the TV after the group has changed, and that seem to be the ones that really appreciated the film, can continue to watch it till the end.

5.2. Foursquare dataseet

Dataset characteristics – Foursquare is a Location-Based Social Network (LBSN) through which users can share their position with friends and talk about visited places, providing comments and recommendations. Foursquare has about 50 million monthly active users. In this paper, we consider the dataset provided in [25, 26] which includes the check-ins (i.e., POI visits) performed by users in 22 months (from April 2012 to January 2014), together with a snapshot of their social connections in that period. This original dataset contains about 22 million check-ins performed by about 110 thousand users on about 4 million venues that have been classified into 20 categories (i.e., entertainment types). Given this dataset, the users' social connections and demographic features (i.e., ages), we built all the information needed for applying the proposed methodology. In particular, we reconstruct approximately 4,000 groups, identified starting from the social connections between users being in the same venue at the same time and belonging to 40 different types, 6,600 sequences of visits regarding 81,000 different POIs, classified into 20 different categories.

Dataset analysis – From a preliminary analysis of the Foursquare dataset, we observed that:

- The average number of POIs contained in each sequence is 2.5.
- The average number of users in each group is 3.8.
- A third of the considered sequences contains at least one short visit (i.e., with a duration less than 10 minutes) and at least one long visit (i.e., with a duration longer than 1 hour).
- In average each sequence with more than 4 venues, contains at least two POIs belonging to the same category.

5.3. Gowalla dataset

Dataset characteristics – Gowalla is another popular LBSN with about 340,000 users. In this paper we consider the dataset provided in [27, 28] which contains 36 million check-ins made by 320 thousand users over about 2.8 million locations belonging to 7 main different categories. From this dataset and the information about users' social connections and demographic features, we reconstruct approximately 296,000 groups, belonging to 25 different types, 111,000 sequences of visits regarding 370,000 POIs classified into 7 main categories. As for the previous dataset, groups are identified starting from the social connections between users being in the same venue at the same time.

Dataset analysis – From a preliminary analysis of the Gowalla dataset, we observed that:

- The average number of POIs contained in each sequence is about 3.5, while the maximum is 14.

- The average number of users in each group is 3.5, while the maximum is 13.

- The 40% of the considered sequences contains at least one short visit (i.e., with a duration less than 10 minutes) and at least one long visit (i.e., with a duration longer than 1 hour).

5.4. Technique evaluation

In order to test the goodness of the proposed approach, we split each dataset into a training set, including 80% of the input sequences, and a test set, containing the remaining ones. The test set is used to assess the provided recommendations and particularly for evaluating the improvement made on the objective functions w.r.t. the initial solutions. The training set is used to both extract all the dynamic information described in Sect. 3, such as the contextual preferences ([D1][D2]), the possible group type evolutions ([D3]), the preference associated to each possible genre transition ([D4]), and to build the initial Pareto-set. Given all these preliminary information, we have applied the proposed technique and compared the obtained results with respect to a base line represented by the work in [8] where: (a) the dynamic aspects [D3] and [D4] are not considered, (b) the only optimization functions are represented by the group satisfaction and the two balancing metrics. Moreover, since [8] does not consider sequence of suggestions, but only single one, we compute the satisfaction and balancing of an experience as the summation of the satisfaction and balancing associated to each single experience composing it.

Each sequence in the test set is firstly used to extract a recommendation query by considering its initial context (group type and time slot), the maximum duration and the available budget; then, it is used to compute the improvement of the proposed solutions w.r.t. all objective functions.

Table 2: Experiment results: the row "baseline" contains the results obtained by considering an adaptation of [8] to sequences of suggestions, while the row 'dynamic" contains the results produced by the technique proposed in this paper.

Dataset	Method	$\cap_{\text{genre}}$	$\uparrow f_d$	$\uparrow f_e$	$\uparrow f_h$	$\uparrow f_s$	$\uparrow f_m$	$\uparrow f_j$
Auditel	baseline	87%	18%	60%	3%	77%	73%	67%
	dynamic	87%	31%	81%	10%	85%	88%	87%
Foursquare	baseline	94%	27%	58%	5%	88%	75%	71%
	dynamic	95%	36%	90%	23%	93%	80%	75%
Gowalla	baseline	89%	24%	63%	6%	72%	78%	82%
	dynamic	91%	29%	91%	31%	75%	81%	85%

Tables 2-3 report for each objective function the improvements provided by both the recommendations produced through the baseline solution and the ones obtained with the proposed technique (i.e., "dynamic"). In Table 2, we evaluate different measures: (a) $\cap_{\text{genre}}$, the percentage of matches between the genre contained in each provided recommendation and the genre included in the corresponding original sequence, (b) the percentage of cases in which the value of an objective function computed on the provided solution is better than the value of the same objective function computed on the corresponding original sequence. Metric (a) confirms that even if the produced suggestions try to maintain a similarity with the starting solution, a certain level of serendipity is also introduced by the technique.

Concerning the value of the objective functions, we can notice that in both cases the provided suggestions are able to improve them w.r.t. the corresponding original sequence. However, since the first three functions are not included in the baseline optimization, they are sightly less affected by the optimization process. The smaller improvement regards the objective function f_h, which deals with the entertainments already enjoyed by users: even if it is an important optimization criterion, its improvement w.r.t. the original sequences is relatively low because the users are inclined alone to prevent enjoying the same entertainment twice, so this avoidance of redundancy is also present in the original sequences. This holds particularly for the TV domain, since in the tourism one, it may happen that the same POI is visited multiple times by the same user.

Another objective function that is only slightly influenced by dynamic aspects is f_d which represents the difference between the duration of the suggested sequence and the duration of the original sequence. In this case, since each query is built by extracting its parameters from an original sequence, including the desired duration, this improvement is likely to be essentially due to the absence of idle time in the middle of a sequence for deciding the next activity to perform.

Regarding to the duration of each single entertainment w.r.t. its effective or suggested duration, namely function f_e, the proposed solution tries to reserve enough time to completely enjoy it (e.g. TV show duration or suggested POI visiting time). Moreover, when a dynamic approach is considered, this value becomes even better. This can be due to the fact that in the considered use case, channels are thematic and uniformity of genres in a sequence can be obtained by choosing subsequent programs in the same channel that clearly have no intersections in their on air intervals. In some measure, the same also holds in the tourist domain, since in some cities, attractions belonging to the same category (e.g., cultural, sport facilities) are placed near each other (e.g., city center or thematic districts).

Finally, considering the last three columns f_s, f_m and f_j, we can register an increment of the group preference and preference balancing in both cases, with a better effect in the dynamic case.

Table 3: Comparison between the relative standard deviation (RSD) computed on the sequences contained in the original test set, and the sequences contained in the recommended sequences obtained by the baseline method and the dynamic approach proposed in this paper.

	% δf_s		% δf_m		% δf_j	
	baseline	dynamic	baseline	dynamic	baseline	dynamic
Auditel	50%	86%	79%	84%	53%	91%
Foursquare	45%	91%	73%	87%	47%	94%
Gowalla	52%	82%	81%	94%	41%	87%

Table 3 illustrates other three important measures: "% δf_s" is the average percentage of improvement of the objective function f_s w.r.t. the original sequence, "% δf_m" is the average percentage of improvement of the objective function f_m and "% δf_r" is the average percentage of improvement of the objective function f_r. From these measures we can observe that, not only the proposed technique is able to increase the value of the objective functions in a greater number of cases, but also that the amount of such improvement is greater than the one produced by the baseline technique.

The obtained results suggest the effectiveness of the proposed approach and will encourage to further investigate the role of dynamic information in the construction of the recommended sequences.

5.5. Scalability Evaluation

We have evaluated the scalability of the proposed technique by comparing the execution time of the annealing procedure in Alg. 10 in three situations: (a) while changing the number of nodes at different values of the initial temperature (see Fig. 3), (b) at different numbers of perturbations for each temperature value (see Fig. 4) and (c) while changing the number of historical input records (see Fig. 5). As you can notice, the use of a MapReduce job for executing the annealing procedure greatly increases the efficiency of the overall work.

When the number of cluster nodes is equal to 1, the performances can be considered the ones of a sequential version of the MOSA algorithm. By considering Fig. 3-5, you can notice that the introduction of a MapReduce version of the MOSA technique greatly improves the performances of the proposed approach. This not only reduces the amount of time required for producing the suggestions, but it also reduces the effect of increasing the complexity of the technique. More specifically, the initial temperature and the amount of perturbations performed on each candidate solution are both elements which influence the complexity of each iteration. In case of a sequential execution, by increasing any of them we obtain an important impact in the performances. Conversely, as you can notice in Fig. 3 and 4, such increment has a lower impact as the number

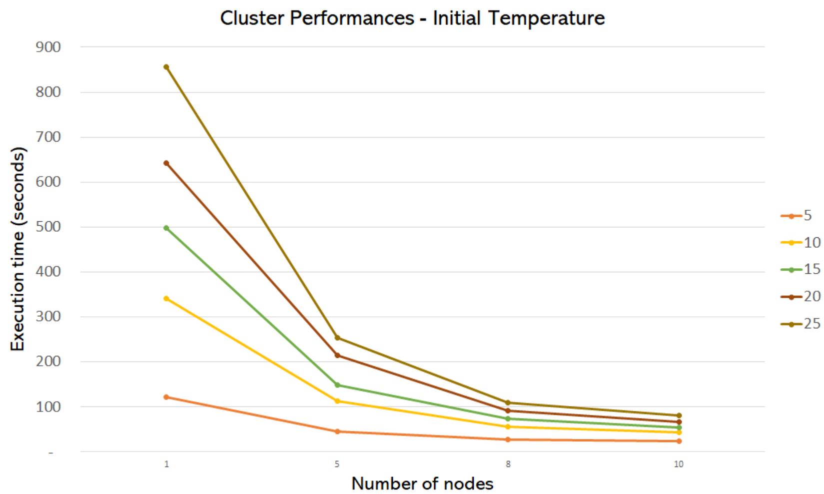

Figure 3: Performances of the MapReduce Job by increasing the number of cluster nodes (from 1 to 10). Each curve is related to a different initial temperature value (from 5 to 25).

of cluster nodes increases.

Finally, we can observe the effect of the parallel execution as the number of input historical sequences changes. Also in this case, the presence of nodes able to work in parallel allows to mitigate the effect of increasing the amount of input information, confirming the good scalability of the proposed approach.

6. Related Work

Recommendation systems represent a well-established research area, and the related literature is therefore vast – see [29, 2] for two comprehensive surveys, and the references therein. Here, we highlight some representative works which focus on specific scenarios, so that to highlight the differences with the problem we study.

Most of the proposed solutions consider a single user, for a single recommendation – see [30] and the references therein. Given the user preferences or the previous interactions, the system automatically suggests the next best choice by minimizing a cost function (or maximing the "utility" for the user). All these

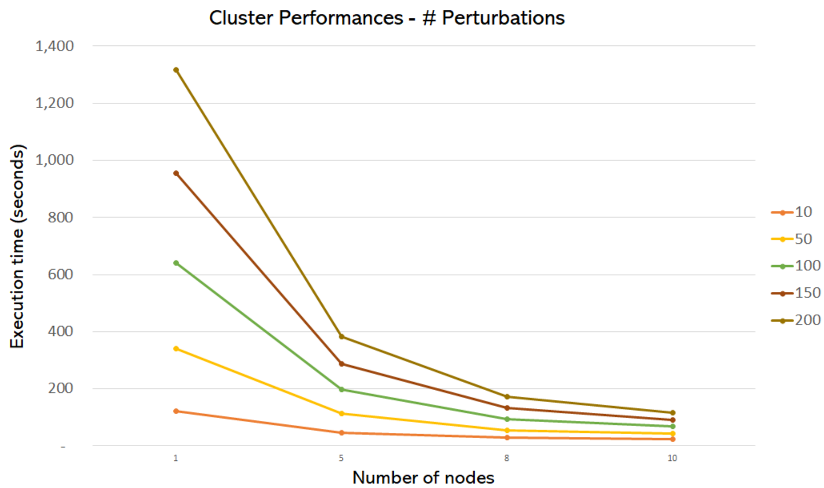

Figure 4: Performances of the MapReduce Job by increasing the number of cluster nodes (from 1 to 10). Each curve is related to a different number of perturbations which is performed on each candidate solution (from 10 to 200).

studies are not easily applicable to the case of sequences, which we consider in our work.

Recommendations can be done also for a group of users that do an activity together, e.g., watching a film [31][13], and in this scenario the system needs to suggest items considering users' collective preferences [2, 3]. In case of a persistent group – a group with consistent structure and historical interactions – the techniques adopted for the single user can be easily extended to this case [32][33]. In case of ephemeral groups, the main problem to solve is to find a suitable measure of the group utility to maximize [7], along with other issues related to the problem of balancing the preferences of different users in a group [8]. Also in these cases, the works do not consider sequences.

If different activities are related together, e.g., visiting points of interest in a city [34], listening to songs [35], it is interesting to provide a recommendation for the whole sequence of such activities. The vast majority of the works on this topic (e.g., [12], [36], [37], [38], [39] and [40]) focus on a single user, also when they take into consideration the influence of context, in particular relatively to

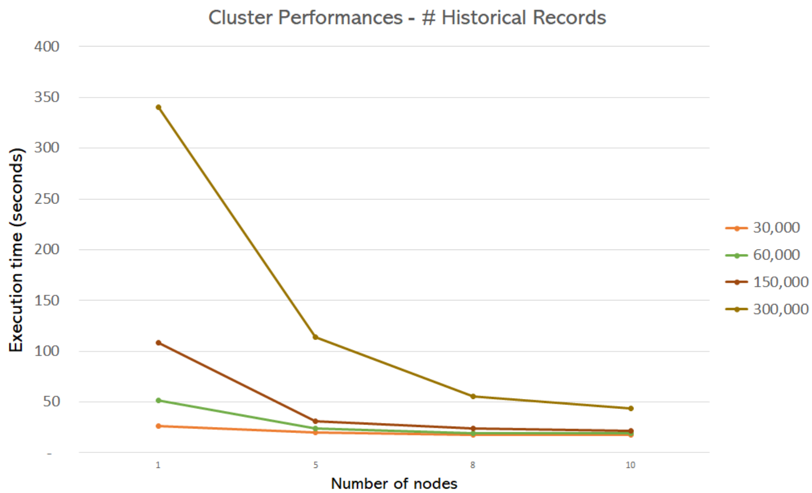

Figure 5: Performances of the MapReduce Job by increasing the number of cluster nodes (from 1 to 10). Each curve is related to a different number of input historical records (from 30,000 to 300,000).

the geographical dimension, neglecting the influence of the group composition on the satisfaction maximization.

In our work, we consider the recommendation of a sequence of activities to a group of users. There are only few papers that study such a scenario [10] [41] [42], and [39]. In [10], the authors provide a system that suggests the path to follow within a museum by a group of visitors. The authors of [41] propose a method for suggesting a sequence of songs to a group of listeners trying to balance the users satisfaction levels. Herzog [42] considers the recommendation of sequences of points of interest for a group of users. All the above works share a common limitation: they consider a single utility function for each user. In our work, instead, we consider the case where the choice is driven by multiple criteria, and formalize the problem as a multi objective optimization problem. In addition, we consider how the context influences the recommendation and we propose a MapReduce implementation to obtain a parallel recommendation computation. Our approach is able to improve the quality of recommendations since it provides a more complex model of the users' preferences.

Finally, another important aspect considered in this paper is the concept of *preference balancing* in group recommendations. Indeed, our main aim is to build a recommendation system for groups able to consider the satisfaction of the individuals composing the group. In particular, we take into consideration the measures of balancing used in [8], which the authors defines as *group fairness measures*, and we integrate them into our dynamic recommendation system. These measures have been properly extended to produce sequence of recommendations, instead of single suggestions.

7. Conclusion

We have presented a system based on a multi-objective optimization algorithm that recommends sequences of activities to dynamic group of users considering their contextual information. The contributions are: (i) the definition of the preferences of an ephemeral group starting from individual preferences and taking into consideration contextual, dynamic and balancing aspects, (ii) the derivation of dynamic and contextual information starting from the historical data usually collected in log files, (iii) the formalization of the problem as a multi-objective optimization problem and (iv) the definition of an efficient MapReduce implementation of the MOSA algorithm in order to provide online suggestions to users, (v) the evaluation of the prosed technique w.r.t. a real datasets.

The proposed recommendation system is able to provide to a group of users a sequence of entertainments that improves the defined objective functions w.r.t. the group choices registered in the logs. In particular, when compared with the baseline method, the proposed approach the dynamic approach provides further improvements in the quality of obtained recommendations in terms of both the number of cases in which the objective functions are improved and the amount of such improvement. Moreover, the performed experiments confirm the good scalability of the MapReduce implementation w.r.t. a sequential execution. Future work includes: (i) the use of explanation techniques for justifying the suggestions to the users and promoting a conscious choice, and (ii) the use of real feedbacks about the actual choices performed by a group among the possible alternatives, to better calibrate future suggestions.

References

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Transactions on Knowledge and Data Engineering 17 (6) (2005) 734–749.

[2] D. Dara, C. R. Chowdary, C. Kumar, A survey on group recommender systems, J. Intell. Inf. Syst. 54 (2) (2020) 271–295. doi:10.1007/s10844-018-0542-3.
URL https://doi.org/10.1007/s10844-018-0542-3

[3] D. Cao, X. He, L. Miao, G. Xiao, H. Chen, J. Xu, Social-enhanced attentive group recommendation, IEEE Trans. Knowl. Data Eng. 33 (3) (2021) 1195–1209. doi:10.1109/TKDE.2019.2936475.
URL https://doi.org/10.1109/TKDE.2019.2936475

[4] M. O'Connor, D. Cosley, J. A. Konstan, J. Riedl, Polylens: A recommender system for groups of user, in: Proc. of ECSCW 2001, 7th European Conference on Computer Supported Cooperative Work, Kluwer, 2001, pp. 199–218.

[5] E. Quintarelli, E. Rabosio, L. Tanca, Recommending New Items to Ephemeral Groups Using Contextual User Influence, in: Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 285292. doi:10.1145/2959100.2959137.
URL https://doi.org/10.1145/2959100.2959137

[6] G. Adomavicius, R. Sankaranarayanan, S. Sen, A. Tuzhilin, Incorporating contextual information in recommender systems using a multidimensional

approach, ACM Transactions on Information Systems 23 (1) (2005) 103–145.

[7] J. Masthoff, Group modeling: Selecting a sequence of television items to suit a group of viewers, User Modeling and User-Adapted Interaction 14 (1) (2004) 37–85.

[8] L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, M. Shaoping, Fairness-Aware Group Recommendation with Pareto-Efficiency, in: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17, Association for Computing Machinery, New York, NY, USA, 2017, p. 107115. doi:10.1145/3109859.3109887.
URL https://doi.org/10.1145/3109859.3109887

[9] D. Sacharidis, Top-n group recommendations with fairness, in: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 16631670. doi:10.1145/3297280.3297442.
URL https://doi.org/10.1145/3297280.3297442

[10] S. Rossi, F. Barile, C. Galdi, L. Russo, Recommendation in museums: paths, sequences, and group satisfaction maximization, Multimedia Tools and Applications 76 (24) (2017) 26031–26055.

[11] H. Liang, K. Wang, Top-k route search through submodularity modeling of recurrent poi features, in: SIGIR '18, ACM, 2018, pp. 545–554.

[12] M. Quadrana, P. Cremonesi, D. Jannach, Sequence-aware recommender systems, ACM Comput. Surv. 51 (4) (2018) 1–36.

[13] J. Masthoff, Group recommender systems: aggregation, satisfaction and group attributes, in: Recommender Systems Handbook, Springer, 2015, pp. 743–776.

[14] S. Migliorini, D. Carra, A. Belussi, Adaptive trip recommendation system: Balancing travelers among pois with mapreduce, in: 2018 IEEE International Congress on Big Data, 2018, pp. 255–259.

[15] S. Migliorini, E. Quintarelli, D. Carra, A. Belussi, Sequences of recommendations for dynamic groups: What is the role of context?, in: 2019 IEEE International Congress on Big Data (BigDataCongress), 2019, pp. 121–128. doi:10.1109/BigDataCongress.2019.00029.

[16] K. Denis, W. Shuaiqiang, V. Jari, A survey of serendipity in recommender systems, Knowledge-Based Systems 111 (2016) 180–192. doi:https://doi.org/10.1016/j.knosys.2016.08.014.
URL https://www.sciencedirect.com/science/article/pii/S0950705116302763

[17] K. Sprck Jones, A statistical interpretation of term specificity and its application in retrieval, Journal of Documentation 28 (1972) 11–21.

[18] R. Konow, W. Tan, L. Loyola, J. Pereira, N. Baloian, A visited item frequency based recommender system: Experimental evaluation and scenario description, Journal of Universal Computer Science 17 (14) (2011) 2009–2028.

[19] R. Jain, D. M. Chiu, W. Hawe, A quantitative measure of fairness and discrimination for resource allocation in shared computer systems, CoRR cs.NI/9809099 (1998).

[20] E. Ulungu, J. Teghem, P. Fortemps, D. Tuyttens, Mosa method: a tool for solving multiobjective combinatorial optimization problems, Journal of Multi-Criteria Decision Analysis 8 (4) (1999) 221–236. doi:https://doi.org/10.1002/(SICI)1099-1360(199907)8:4<221::AID-MCDA247>3.0.CO;2-O.

[21] D. M. Curry, C. H. Dagli, Computational complexity measures for many-

objective optimization problems, Procedia Computer Science 36 (2014) 185–191. doi:https://doi.org/10.1016/j.procs.2014.09.077.

[22] B. Suman, Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem, Computers & Chemical Engineering 28 (9) (2004) 1849–1871.

[23] E. Aarts, J. Korst, Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing, John Wiley & Sons, Inc., New York, NY, USA, 1989.

[24] Z. J. Czech, Three parallel algorithms for simulated annealing, in: Parallel Processing and Applied Mathematics, Springer Berlin Heidelberg, 2002, pp. 210–217.

[25] D. Yang, B. Qu, J. Yang, P. Cudre-Mauroux, Revisiting User Mobility and Social Relationships in LBSNs: A Hypergraph Embedding Approach, in: The World Wide Web Conference, WWW '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 21472157. doi:10.1145/3308558.3313635.
URL https://doi.org/10.1145/3308558.3313635

[26] D. Yang, B. Qu, J. Yang, P. Cudre-Mauroux, LBSN2Vec++: Heterogeneous Hypergraph Embedding for Location-Based Social Networks, IEEE Transactions on Knowledge and Data Engineering (2020) 1–1doi:10.1109/TKDE.2020.2997869.

[27] Y. Liu, W. Wei, A. Sun, C. Miao, Exploiting geographical neighborhood characteristics for location recommendation, in: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM '14, Association for Computing Machinery, New York, NY, USA, 2014, p. 739748. doi:10.1145/2661829.2662002.
URL https://doi.org/10.1145/2661829.2662002

[28] X. Liu, Y. Liu, K. Aberer, C. Miao, Personalized Point-of-Interest Recommendation by Mining Users' Preference Transition, in: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, CIKM '13, Association for Computing Machinery, New York, NY, USA, 2013, p. 733738. doi:10.1145/2505515.2505639.
URL https://doi.org/10.1145/2505515.2505639

[29] F. Ricci, L. Rokach, B. Shapira, Recommender systems: introduction and challenges, in: Recommender Systems Handbook, Springer, 2015, pp. 1–34.

[30] X. Ning, C. Desrosiers, G. Karypis, A comprehensive survey of neighborhood-based recommendation methods, in: Recommender Systems Handbook, Springer, 2015, pp. 37–76.

[31] A. Jameson, B. Smyth, Recommendation to groups, Springer, 2007, pp. 596–627.

[32] I. Ronen, I. Guy, E. Kravi, M. Barnea, Recommending social media content to community owners, in: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, ACM, 2014, pp. 243–252.

[33] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, W. Cao, Deep modeling of group preferences for group-based recommendation., in: AAAI, Vol. 14, 2014, pp. 1861–1867.

[34] Z. B. Sojahrood, M. Taleai, A POI group recommendation method in location-based social networks based on user influence, Expert Syst. Appl. 171 (2021) 114593. doi:10.1016/j.eswa.2021.114593.
URL https://doi.org/10.1016/j.eswa.2021.114593

[35] Z. Cheng, J. Shen, L. Zhu, M. Kankanhalli, L. Nie, Exploiting music play sequence for music recommendation, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI'17, 2017, pp. 3654–3660.

[36] S. T., A. S., A. K., Selecting sequences of items via submodular maximization, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI Press, 2017, pp. 2667–2673.

[37] R. Srivastava, S. Hingmire, G. K. Palshikar, S. Chaurasia, A. Dixit, CSRS: a context and sequence aware recommendation system, in: Proceedings of the 8th annual meeting of the Forum on Information Retrieval Evaluation, ACM, 2016, pp. 8–15.

[38] J. Liu, Z. Zhang, C. Liu, A. Qiu, F. Zhang, Exploiting two-dimensional geographical and synthetic social influences for location recommendation, ISPRS Int. J. Geo Inf. 9 (4) (2020) 285. doi:10.3390/ijgi9040285.
URL https://doi.org/10.3390/ijgi9040285

[39] J. Huang, Y. Liu, Y. Chen, C. Jia, Dynamic recommendation of POI sequence responding to historical trajectory, ISPRS Int. J. Geo Inf. 8 (10) (2019) 433. doi:10.3390/ijgi8100433.
URL https://doi.org/10.3390/ijgi8100433

[40] Y. Ma, J. Mao, Z. Ba, G. Li, Location recommendation by combining geographical, categorical, and social preferences with location popularity, Inf. Process. Manag. 57 (4) (2020) 102251. doi:10.1016/j.ipm.2020.102251.
URL https://doi.org/10.1016/j.ipm.2020.102251

[41] A. Piliponyte, F. Ricci, J. Koschwitz, Sequential music recommendations for groups by balancing user satisfaction, in: UMAP workshops, 2013.

[42] D. Herzog, Recommending a sequence of points of interest to a group of users in a mobile context, in: Proceedings of the eleventh ACM conference on recommender systems, ACM, 2017, pp. 402–406.