Sequences of Recommendations for Dynamic Groups: What is the Role of Context?

Sara Migliorini
Università di Verona
Verona, Italy
sara.migliorini@univr.it

Elisa Quintarelli
Università di Verona
Verona, Italy
elisa.quintarelli@univr.it

Damiano Carra
Università di Verona
Verona, Italy
damiano.carra@univr.it

Alberto Belussi
Università di Verona
Verona, Italy
alberto.belussi@univr.it

***Abstract*—Recommendation algorithms have been investigated and employed by many important companies in the past years: some scenarios, such as the one where a system suggests the points of interest to tourists, well adapt to sequence of recommendations to (groups of) users. We envision that sequence recommendations can be useful whenever the group of users has a limited time interval to spend together, since they reduce the time wasted in selecting the best next activity.**

In this paper, we investigate the role played by the context, i.e. the situation the group is currently experiencing, in the design of a system that recommends sequences of activities. We model the problem as a multi-objective optimization, where the satisfaction of the group and the available time interval are two of the functions to be optimized. In particular, the dynamic evolution of the group can be considered as the key contextual feature to produce better suggestions.

***Index Terms*—Sequence recommendations, Optimization Problem, Context, Groups**

I. Introduction

Nowadays users are more and more often choosing what interests them on-line in huge catalogues: prominent examples are the lists of available video-on-demand movies and series, songs, restaurants nearby the user, or the products that can be purchased in online stores. These long lists of choices, if not properly managed and personalized, risk to confuse users instead of being a precious resource. Recommender systems (see [1] for a survey) suggest items that match the tastes of users and have quickly become a fundamental technology, employed by many important companies such as Netflix, Amazon and Google. In all the scenarios where activities are inherently social (e.g. going to the cinema, eating out or visiting a city) recommendation systems provide suggestions for **groups** of users, which in the literature are classified either as persistent or ephemeral [2]. Persistent groups are those where members have a history of activities together, and thus can be considered as a normal user. Ephemeral groups, on the other hand, may be constituted by people who are together for the first time: consequently, the group preferences have to be computed on the basis of similar persistent groups, where group similarity is evaluated by means of common features defining the context of the groups.

The online recommendation of the next activity, either for single users, or for groups, has been studied extensively [1] [2]: we refer to this type of recommendation as *myopic*, since it focuses on one activity at a time. Such an approach does not consider that further recommendations could produce a better satisfaction of the group preferences by extending the evaluation to an interval of time (optimization problem). 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 tourist [3], [4] or the list of songs to listen [5], in which planning ahead the recommendation of a whole sequence – given some constraints, such as the available time – provides more flexibility. Moreover, it gives the chance to find solutions that a myopic recommendation may not able to find. Nevertheless, most of the works for sequence recommendations focus on a single user and do not consider groups (see the survey [5]).

In this paper we consider the problem of recommending a sequence of activities to groups of users, either persistent or ephemeral. The resulting sequence could reduce the thinking time required by a myopic recommendation system – e.g., channel surfing that users typically perform to find something interesting to watch next, or discussions during a trip to decide the next thing to visit. Recommending sequences can be useful whenever the group of users has a limited time interval to spend together, since suggesting a sequence decreases the time wasted in selecting the best next activity. In addition, we consider the role played by the **context**, i.e. the situation, the group is currently experiencing. It has been recognized that the notion of context influences the user preferences [6], which may vary with respect to the group current context. For instance, the same person, may prefer to see different film genres depending on the people she/he is with (e.g., family, or friends), or depending on the day (weekdays, weekend), or on period of the year (e.g., Christmas time).

Our approach proposes to model the generation of recommendation sequences as a multi-objective optimization problem, where the satisfaction of the group is one of the functions to be optimized. We propose to solve such problem by using the Multi-Objective Simulated Annealing (MOSA) optimization heuristic [7]. More specifically, we extend such approach with the notion of context. The novelty with respect to the state of the art (see [5]) is that time (a contextual feature) influences the group composition, thus the current context. We envision that, given the sequence recommendation problem, time has a great impact in the evolution of groups and has a central role in the optimization of user/group preferences.

Contributions: We address the following main issues.

- *Recommendations:* We study the problem of recommending a sequence of activities to groups of users by considering their current context, and we formulate it as multi-objective, optimization problem.
- *MOSA:* We solve the problem with a distributed, multi-objective Simulated Annealing, implemented in MapReduce, so that to rapidly provide suggestions to groups.
- *Evaluation:* We evaluate our solution using real-world traces of user activities, and show that we are able to provide better suggestions with respect to the choices made by the users.

Roadmap: We provide the motivation in Sect. II and define the problem in Sect. III. We present our solution in Sect. IV, and we evaluate it in Sect. V. Sect. VI discusses the related works, and we conclude the paper in Sect. VII.

II. Context and Motivation

The proposed approach is general enough to be applied to any activity, as formalized in Sect. III-B. In this section, due to the availability of a dataset related to the TV domain, we consider the entertainment scenario as our motivating example. The issues that should be considered are the following:

Sequences are produced for groups evolving along with time. We are interested in generating sequences of recommendations for people spending a time interval together. The composition of the group, which can change over time, is a feature that influences the recommendations, since movies to be suggested to adults are not always proper for kids. We consider as contextual features the group composition (*adults with kids*, *adults*, *teens*, or *kids*) and the temporal information (*daytime* or *night*, and the day of the week), but the proposal can be easily extended by including other relevant features to describe the contexts users may experience in a given scenario. We say that a group has changed whenever its current composition changes, i.e. from *adults with kids* the group has evolved into *adults* (no matter the number of members): our recommendations are contextual, thus, a change in context determines a different policy in providing recommendations to the actual group. Note that the group evolution is natural along with time, indeed, a group can dynamically change its composition; thus, a contextual feature (in this case the time) can have an impact on another feature (the group composition).
Problem to solve: In order to optimize recommendation sequences for a group in a given time interval, it is necessary to investigate how time can influence the group composition.

Constraints and objective functions are considered in exploring the problem. As we will describe later, MOSA is a multi-objective optimization technique that can take into account various constraints to be satisfied and functions to be optimized: different alternatives can be considered in aggregating individual preferences to obtain the group recommendation. In our scenario, we will consider as constraints the maximum interval of time the group can spend together ($T_{\max}$) and the maximum available budget ($b_{\max}$). As functions to optimize we will consider: i) the minimization of the empty slot between two following TV programs; ii) the maximization of the portion of $T_{\max}$ covered by recommendations; iii) the minimization of the number of programs the group components has already watched in the past; iv) the maximization of the group satisfaction. Notice that the latter is influenced not only by the context, but also by the possible evolution of the group.
Problem to solve: Find the set of functions that can best describe the scenario we consider.

Customized sequences of recommendation for groups. We analyze different possibilities to aggregate individual preferences to generate group preferences. To compute the preferences of a group for a given program we use the classical aggregation functions, i.e. the Average Preference or the Least-Misery Preference. However, the analysis of past evolutions of groups can allow us to predict the evolution of the current group over time, e.g. it is frequent that after 9 p.m the group changes from *adults with kids* to *adults*; in this case we try to maximize the satisfaction of kids inside the group in the early phase, since we know that with high probability the group will evolve into another one with a different composition.
Problem to solve: Consider the impact of the evolution of the group in providing the recommendation.

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 optimatzion problems, still the computational complexity remains high. For this reason, the proposed solution should be implemented with an approach that allows for parallel computation, so that to keep the time necessary to provide the recommendation as short as possible.
Problem to solve: Provide a solution that is computationally efficient and generate the recommendation quickly.

III. A Recommendation System for Groups

A. System overview

Figure 1 shows an overall picture of our approach (in the blue rectangles the run-time steps): we collect the description of items of our scenario, e.g. different kind of entertainments, and the logs of past accesses both for users and groups. When a group is ephemeral, its preferences will be inferred starting from the components' preferences or by using the preferences of a similar (w.r.t. the context) group. The collected information about the past users/groups history, can be used to predict a group evolution with a certain probability.

The context is related to the group composition (*adults with kids*, *adults*, *teens*, or *kids*) and temporal information (*daytime* or *night*, and the day of the week), but it may include other features. Given a group in a certain context (the temporal information can be sensed, while the group composition has to be declared) and the slot of time to spend together, our

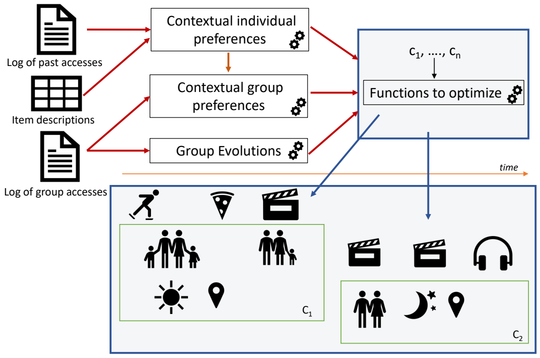

Fig. 1: Our approach: a high level vision

framework analyzes the preferences computed off-line and determines a sequence of suggested entertainments. These sequences are affected also by the possible group evolution computed using the past known history. In particular, if the current group is a persistent one, its past history can be used to predict its evolution with an high degree of confidence; otherwise, we can use the past history of similar groups in order to make such prediction also for an ephemeral group.

We now discuss how the problem can be formulated as an optimization problem (Sect. III-B) and then we show our solution based on the MOSA technique (Sect. IV).

B. Problem Formulation

This section provides a first formalization of the problem of suggesting a sequence of recommendations to s.

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 the duration $e.dur$, the genre $e.gen$, the suggested age $e.age$, and a cost $e.cost$.

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

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 entertainments O, the set of all possible experiences, denoted by $\mathcal{E}$, contains all possible ordered combinations of entertainments in O, for any cardinality of ε.

Let $\varepsilon = \langle e_1, \ldots, e_n \rangle$ be an experience, its overall duration $\delta(\varepsilon)$ is defined as $\delta(\varepsilon) = \sum_{i=1}^{n} e_i.dur$; while its overall cost $\gamma(\varepsilon)$ is defined as $\gamma(\varepsilon) = \sum_{i=1}^{n} e_i.cost$.

Definition 3 (Dynamic Group): Given a set of users $\mathcal{U}$, a dynamic group g is a function that associates to each time instant t a subset $\{u_1, \ldots, u_n\} \subseteq \mathcal{U}$. Each (dynamic) group has a type $\overline{g}$, which is dynamic too, thus it depends on t.

The notion of group considered in this paper is a collection of users whose composition evolves with the time. Notice that, besides temporal information, we are interested in the group composition for determining the context (e.g., *adults with kids* or *adults*), not in the specific individual users that are currently inside the group.

Definition 4 (Dynamic Context): A dynamic context c is characterized by the type of a dynamic group composition $\overline{g}$ (e.g. *adults*) and a temporal information τ (e.g. *daytime* or *night*, and the day of the week).

As previously discussed, the possible evolution about the group composition can be determined using the past collected history. Moreover, such evolution can be used to influence the computation of a group preferences in a given context.

Definition 5 (Group Preference): Let $c = \langle \overline{g}, \tau \rangle$ be a dynamic context of a group g, e_j be an entertainment and t_i a temporal instant. The function $\bar{p}(g, c, t_i, e_j)$ computes the preference of the group g for the entertainment e_j in the context c considering its composition at the time t_i and its possible evolution at time t_{i+1}.

Proposition 6: Let $c = \langle \overline{g}, \tau \rangle$ be a dynamic context of a group g and t_i a time instant, such that $g(t_i) \cap g(t_{i+1}) \neq \emptyset$, and let 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 $\bar{p}(g, c, t_i, e_j)$.

The rationale behind the above proposition is that the users that will leave the group have no chance to be satisfied in the future, while the users that remain in the group can be satisfied also by the next suggestions.

Definition 7 (Recommendation query): A group of users looking for a recommendation submits a query Q to the system containing the following information:

- the initial context c_0: it includes at least the group composition, the date and time at which the experience will start;
- the desired duration as an interval $(d_{\min}, d_{\max})$;
- the mandatory maximum duration $T_{\max}$;
- the mandatory maximum available budget $b_{\max}$;

Notice that, while the start time t_0, the maximum duration $T_{\max}$ and the maximum budget $b_{\max}$ are considered as mandatory constraints, the desired duration is intended as a desiderata: the recommended experiences can have an overall duration close to the desired one.

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

Definition 8 (Objective functions): Given a recommendation query Q, an experience $\varepsilon = \langle e_1, \ldots e_n \rangle$ and a a dynamic context $c = \langle \overline{g}, \tau \rangle$, the considered objective functions to be minimized are:

- $f_d(\varepsilon, c) = \begin{cases} w_a \cdot (d_{\min} - \delta(\varepsilon)) & \text{if } \delta(\varepsilon) < d_{\min} \\ w_b \cdot (\delta(\varepsilon) - d_{\max}) & \text{if } \delta(\varepsilon) > d_{\max} \\ d_{\max} - \delta(\varepsilon) & \text{otherwise} \end{cases}$
  f_d computes the difference between the actual duration of the experience ε and the desired duration, i.e. it is essentially as a measure of the empty slots. The two weights w_a and w_b can be used to consider less serious a duration less than the minimum desired one, w.r.t. a duration greater than the maximum desired one.
- $f_e(\varepsilon, c) = \sum_{e_j \in \varepsilon} \mid e_j.dur - (e_j.end - e_j.start) \mid$
  f_e computes the sum of the portions of entertainments the user (or group) did not enjoy, which has to be minimized.

- $f_h(\varepsilon, c, t_i) = |\bigcup_{u_i \in g(t_i)} H(u_i) \cap \bigcup_{e_j \in \varepsilon} e_j|$
  f_h counts the number of entertainments already viewed by each components of the group, considering its past viewing. $H(u_i)$ is the past history of the user u_i, i.e. it is the set of entertainment enjoyed by u_i in the past. Function f_h depends on the time instant t_i in which it is computed.
- $f_s(\varepsilon, c, t_i) = n \cdot S - \sum_{i=1}^{n} \bar{p}(g, c, t_j, e_j)$
  The function f_s minimizes the loss of preference, where S is the possible maximum degree of satisfaction (preference) for an entertainment, $\bar{p}(g, c, t_j, e_j)$ as been defined in Def. 5.

The group recommendation problem can be formulated as an optimization problem:

$$\begin{aligned} \underset{\varepsilon}{\text{Minimize}} \quad & \langle f_d, f_e, f_n, f_s \rangle \\ \text{subject to} \quad & \delta(\varepsilon) < \mathrm{TD}_{\max} \\ & \gamma(\varepsilon) < b_{\max} \end{aligned} \tag{1}$$

The four objective functions can be combined in a single function $\bar{f} : \mathcal{E} \to \mathbb{R}^4$ 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.

IV. Proposed Solution: a MOSA Approach

In the previous section we have defined the group recommendation problem as an optimization problem. However, finding an exact solution for Eq. 1 is computational expensive. For this reason, we need to apply some well known heuristics in order to produce a solution in a reasonable amount of time. Among all possible optimization heuristics, we choose to apply the Multi-Objective Simulated Annealing (MOSA) [8] technique. The rational behind this choice is twofold: (1) it has been demonstrated its ability to reach a global optimum if annealed sufficiently slow [9], while other solutions, like the greedy algorithm, can suffer from the problem of getting stuck in local optima; (2) rather than using a random artificial solution as starting point in the search space exploration, we can rely on the available historical data, concerning the past experiences performed by the same group (if available), or by similar groups, in the given context.

The idea under the MOSA technique is that at each step of the procedure, the current solution s_1 is sightly modified in some way (perturbed) obtaining a new solution s_2, then s_2 is chosen in place of 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 happen in physical annealing procedures. The mentioned perturbation operation is the way used to explore the solution space. In the considered problem, example of perturbations are (i) the application of elementary *changes* to the current solution, such as the removal or addition of a single entertainment, (ii) the *replacement* of an entertainment with a new one, or (iii) the change in the *order* of the entertainments.

With multi-objective optimization, we can define a partial order on the solutions based on the concept of dominance. Experiences $\varepsilon_{\text{curr}}$ and ε_{new} are mutually non-dominating if and only if neither dominates the other one. The set of mutually non-dominating solutions is called *Pareto-set*, and it is denoted by $\mathcal{PS}$. A solution not dominated by any other solution is called *Pareto-optimum*. From the Pareto-set $\mathcal{PS}$ we can compute the *Pareto-front* $\mathcal{F} \subseteq \mathbb{R}^4$, which is the image of $\mathcal{PS}$ in the objective space: $\mathcal{F} = \{\bar{f}(\varepsilon) \mid \varepsilon \in \mathcal{PS}\}$.

The goal of a MOSA algorithm is to start from a initial set of solutions, the current *Pareto-set*, compute the corresponding current Pareto-front and move it towards the optimal Pareto-front (the Pareto-front of the Pareto-optimum set) while encouraging the diversification of the candidate solutions. In particular, the probability of making a transition from the current solution $\varepsilon_{\text{curr}}$ towards a new solution ε_{new} is specified by an acceptance probability function $P(\varepsilon_{\text{curr}}, \varepsilon_{\text{new}}, \mathcal{C})$ which depends upon the global parameter $\mathcal{C}$ (*temperature*) and the energy of the two solutions. The energy of a solution ε, denoted by $E(\varepsilon, \mathcal{F})$, measures the portion (number of solutions) of the current Pareto-front that dominates ε, i.e., $E(\varepsilon, \mathcal{F}) = |\{v \in \mathcal{F} \mid v \prec \bar{f}(\varepsilon)\}|$. Trivially, a solution with more energy will be accepted with probability 1, while a solution with less energy has also a probability, but less than 1, to be accepted, according to the parameter $\mathcal{C}$ (the higher the temperature, the higher the probability).

The result produced by the application of the MOSA technique to the entertainment problem, called Entertainment MOSA algorithm (or EMOSA), is a set of equally "good" suggestions, which can be proposed to the group as possible alternative experience (sequences of entertainments). A feedback about the actual choice performed by the group, can be used to better calibrate the algorithm for future queries.

The EMOSA has been implemented in MapReduce. In particular, the traditional MOSA algorithm has been subdivided into two MapReduce jobs: the first one, reported in Algorithm 1, initializes the Pareto-set $\mathcal{PS}_{\text{init}}$ using the past collected experiences which satisfy the given recommendation query Q. The Pareto-set produced by this job will be used as starting point for the second job illustrated in Algorithm 2, which actually implements the MOSA heuristic in order to produce the required recommendations. In Algorithm 1, each mapper receives a portion of the historically collected experiences and updates its portion of the initial Pareto-set by considering each of them at time. The reducer essentially combines the portions of Pareto-set produced by the mappers, by removing dominated solutions. The value k is a dummy key used during the job, it can be for instance an identifier for the query Q. The presence of a unique reduce is not a great limitation because the majority of the work has already be done by the mappers.

The MapReduce job in Algorithm 2 computes the desired recommendations given a recommendation query Q, and in particular the initial context c_0, the desired duration

Algorithm 1 MapReduce job for the initialization of the Pareto-set $\mathcal{PS}_{\text{init}}$. Q is the recommendation query.

Mapper

procedure SETUP
  $\mathcal{PS}_{\text{map}} \longleftarrow \emptyset$
end procedure

procedure MAP(*id*, ε)
  if ε satisfies Q **then**
    $\mathcal{PS}_{\text{map}} \longleftarrow$ REMOVEDOM($\mathcal{PS}_{\text{map}}, \varepsilon$)
  end if
end procedure

procedure CLEANUP
  return $(k, \mathcal{PS}_{\text{map}})$
end procedure

Reducer

procedure REDUCE(k, $\langle \mathcal{PS}_1, \mathcal{PS}_2, \dots \rangle$)
  $\mathcal{PS}_{\text{init}} \longleftarrow \emptyset$
  for $\mathcal{PS}_i \in \langle \mathcal{PS}_1, \mathcal{PS}_2, \dots \rangle$ **do**
    for $\varepsilon \in \mathcal{PS}_i$ **do**
      $\mathcal{PS}_{\text{init}} \longleftarrow$ REMOVEDOM($\mathcal{PS}_{\text{init}}, \varepsilon$)
    end for
  end for
  return $(k, \mathcal{PS}_{\text{init}})$
end procedure

Algorithm 2 MapReduce job for the execution of the EMOSA algorithm.

Mapper

procedure MAP(*id*, ε)
  $\mathcal{PS}_{\text{map}} \longleftarrow \emptyset$
  $\mathcal{PS}_{\text{init}} \longleftarrow$ retrieve from cache
  $\mathcal{PS}_{\text{map}} \longleftarrow$ EMOSA($\mathcal{PS}_{\text{init}}, \varepsilon, Q, t_{\text{init}}$)
  return $(k, \mathcal{PS}_{\text{map}})$
end procedure

Reducer

procedure REDUCE(k, $\langle \mathcal{PS}_1, \mathcal{PS}_2, \dots \rangle$)
  $\mathcal{PS} \longleftarrow \emptyset$
  for $\mathcal{PS}_i \in \langle \mathcal{PS}_1, \mathcal{PS}_2, \dots \rangle$ **do**
    for $\varepsilon \in \mathcal{PS}_i$ **do**
      $\mathcal{PS} \longleftarrow$ REMOVEDOM($\mathcal{PS}, \varepsilon$)
    end for
  end for
  return $(k, \mathcal{PS})$
end procedure

Algorithm 3 EMOSA algorithm.

1: **procedure** EMOSA($\mathcal{PS}_{\text{init}}$, ε, Q, t_{init})
2:   $\mathcal{PS} \longleftarrow \mathcal{PS}_{\text{init}}$
3:   $\mathcal{F} \longleftarrow$ COMPUTEPARETOFRONT($\mathcal{PS}$)
4:   $t \longleftarrow t_{\text{init}}$
5:   **while** $t > t_{\min}$ **do**
6:     $\varepsilon' \longleftarrow$ PERTURB(ε, $T_{\max}$)
7:     $\mathcal{F}' \longleftarrow \mathcal{F} \cup \bar{f}(\varepsilon')$
8:     $\Delta_E \longleftarrow$ COMPUTEENERGYDIFF($\varepsilon', \varepsilon, \mathcal{F}'$)
9:     $P \longleftarrow \min(1, \exp(-\Delta_E/T))$
10:     **if** *rand*$(0,1) < P$ **then**
11:       REMOVEDOMINATED($\mathcal{PS}, \varepsilon', \mathcal{F}, \bar{f}(\varepsilon')$)
12:       $\mathcal{PS} \longleftarrow \mathcal{PS} \cup \varepsilon'$
13:       $\mathcal{F} \longleftarrow \mathcal{F} \cup \bar{f}(\varepsilon')$
14:       $\varepsilon \longleftarrow \varepsilon'$
15:     **end if**
16:     UPDATETEMPERATURE(t)
17:   **end while**
18:   **return** $\mathcal{PS}$
19: **end procedure**

$(d_{\min}, d_{\max})$, the mandatory maximum duration $T_{\max}$ and the mandatory maximum budget $b_{\max}$. The initial Pareto-set $\mathcal{PS}_{\text{init}}$ is the one computed by the previous job, while the parameter t_{init} is the initial temperature used by the EMOSA algorithm.

The EMOSA is illustrated in Algorithm 3: given a current experience ε and the initial Pareto-set, it essentially performs some perturbations on ε obtaining a new experience ε'. Than the goodness of ε' w.r.t. ε is evaluated by computing the cumulative objective function $\bar{f}$ as discussed in Definition 8. In line 8, the function COMPUTEENERGYDIFF, computes the energy (goodness) of the two solutions in terms of the number of solutions in the estimated Pareto-front $\mathcal{F}$ that are dominated by each of them. This energy value is used to compute the probability P to choose the new experience ε' in place of the current one ε as new solution. In case ε' is preferred, the Pareto-set has to be updated accordingly. Clearly, the computation of the cumulative objective function $\bar{f}$ needs to access some additional data, such as the user preferences, the duration of each entertainment and so, but we omit them from Algorithm 3 for not cluttering the presentation. In particular, the computation of function f_s is the dynamic component of the algorithm, which requires in input an additional source of information representing the predictions about the group type evolution. Indeed, when the possible group evolution is taken into consideration, the preference of each component can be weighted in different ways depending on the probability he/she will still remain in the group. Another dynamic component that has to be considered is the history about the past views performed by each user in the group, represented by the variable H in function f_h. Clearly, this parameter has to change in time and w.r.t. the current sequence of suggestions that is being produced, for not suggesting the same entertainment twice inside a given experience.

V. EVALUATION

Available dataset. We consider a dataset in the TV domain, with users watching TV programs possibly in groups and in different contexts. Contexts are identified by temporal information (*daytime* or *night*, and the day of the week) and the type of groups, i.e. the number and age of members. The dataset contains TV viewing information related to almost 8,000 users and 119 channels, broadcast both over the air and by satellite. The dataset is composed of an electronic program guide (EPG) containing the description of 21,194 distinct programs (their genre is mentioned in the dataset), and a log containing both individual and group viewings performed by the users. The log contains approximately 5 million entries, among which we retained just the synthonizations longer than three minutes. Almost 1.5 million viewings involve more than one person together. Each log row specifies the identifier of the user and that of the program he/she watched, along with start time and end time.

Dataset analysis. From the analysis of the dataset we found out that: i) the average number of programs each user watches sequentially in a day is about 3; ii) in 80% of cases where there is a sequence of program viewings for a user in a day, the number of short viewings 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; iv) in 32% of cases where there is a sequence of viewings for the same user in a specific day, there are at least two long viewings with short viewings in the middle. The last analysis suggested us the importance of providing sequences of recommendations also in the TV domain, since it could reduce the channel surfing to find something interesting to watch next.

Fig. 2 compares a sequence contained in the original dataset and a corresponding suggestion produced by the EMOSA algorithm. Both sequences take as initial context a group of two adults and two children, in a day time period starting at 15:30 and ending at 21:30. For each sequence in Fig. 2, we report both the involved tv shows with their genre (represented by an icon) and their duration (see the time stamps above and the straight vertical lines), and the effective views performed by the group with its duration (see the time stamps below and dashed lines). In particular, the blue segments denote the duration of each single viewing, as is noticeable in the original sequence, the group performs some short viewings between one tv show and the other (i.e., surfing activity, searching for the next tv show to view). Moreover, in the original sequence the first two chosen shows (the sport event and the film) overlap (see the mixed colours between 16:00 to 16:30). The original sequence is comprised of 4 TV shows overall: the first one is a sporting event the group starts to view after 10 minutes of channel surfing and stops to view before its end; conversely, the members in the group start to watch the following film after it has already started, but they watch until the end. From this behaviour we can conclude that the group would have seen the film from the beginning if they had known about it; on the contrary they had less interest about the TV show, since they watched it only for a small fraction.

The proposed sequence produced by the EMOSA algorithm is reported in the line below: it clearly removes all short views between the long ones; moreover, no loss is registered in the view of the two central tv shows. In particular, the film contained in the original sequence can now be completely watched. Moreover, the prediction about the potential group evolution (namely the fact the children will leave the group after the 21:30), led to favour their preferences and this results into the inclusion of a cartoon after the film, which also can be entirely watched by the group. Additionally, considering this possible group evolution, the documentary is left at the end of the required period, since it is mostly preferred by the adults who are more likely to remain in the future group (eventually continuing to watch the show until its end).

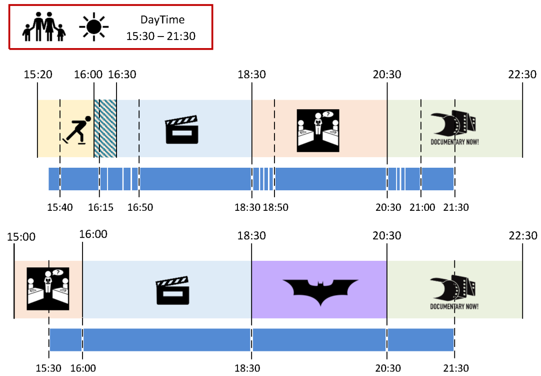

Fig. 2: Comparison of two sequences: one contained in the original dataset and one produced by the EMOSA algorithm.

Experimental results. The experiments on the dataset were performed as follows. The group viewings were split into a training set, including the syntonizations between December 3rd, 2013 and February 15th, 2014 (1,210,316 entries), and a test set, containing the remaining ones (238,748 entries) and used to asses the provided recommendations. From the training set we have learnt the contextual preferences (i.e. preferences correlated to temporal information) about the program genre for both single users and groups and the most likely evolutions of each group type. As described in Definition 3, the type $\bar{g}$ of a group g is given by the ages of its components, therefore an evolution is registered when one or more members leave or join the group changing the overall set of covered ages.

We have computed sequences of n recommendations for all groups (n depends on the duration of the time interval the members in the group spend together). The sequences were computed in two ways: 1) without considering the group evolution along with time 2) by taking into account the information about the group evolution. In particular, the sequences composing the test set are used to defined the various recommendation queries: initial context (group type and time slot), the maximum duration, and so on. Then the EMOSA algorithm has been applied by firstly ignoring the type evolution information and secondly by considering also the computed predictions about the possible evolutions. Tables I-II illustrate the goodness of the suggestions in both cases and a comparison between the recommended sequences and the original ones. More specifically, Table I reports both the percentage of matches between the genre contained in the provided recommendations and the genre included in the test sequences (column $\cap_{\text{genre}}$), together with the improvement in the objective functions induced by the EMOSA algorithm (columns f_d, f_e, f_h and f_s), namely the percentage of cases in which the value of the corresponding objective function is improved w.r.t. to its value in the original sequence.

We notice that the suggested recommendations improve one or more objective functions w.r.t. to the corresponding original sequence. The less improved objective function is f_d

TABLE I: Experiment results: the row "w/o ge" contains the results considering the prediction about the group evolution, while the row "with ge" contains the results obtained by taking into account also the group evolution.

	$\cap_{\text{genre}}$	$\uparrow f_d$	$\uparrow f_e$	$\uparrow f_h$	$\uparrow f_s$
w/o ge	80%	12%	70%	8%	62%
with ge	78%	13%	71%	10%	75%

which represents the difference between the duration of the suggestion and the duration of the original sequence. This is due to the way the recommendation query is built: each query has been built considering the characteristics of a sequence in the test set, therefore, it likely has exactly the right mandatory duration given in the query. The improvement is essentially due to the absence of the short views (shorter than 3 minutes).

Conversely, as regards to the duration of each single view w.r.t. its effective duration, namely function f_e, the proposed solutions prevent in most cases the loss of parts of the TV show, favouring its complete viewing. As regards to f_h, even if it is an important optimization criterion, its improvement w.r.t. to the original sequences is relatively low because the users are inclined alone to prevent the viewing of the same program twice, so this avoidance of redundancy is also present in the test sequences. Finally, as regards to column f_s, we can register an increment of the group preference in both cases, but as will be clear from parameter "RSD f_s" this global improvement is achieved in a different way when considering, or not, the group type evolution. Remember that taking into account the group evolution while computing the value of f_s means that the individual preferences are weighted w.r.t. to the probability they will leave the group in the near future.

Table II contains a comparison between the average relative standard deviation (RSD) of two objective functions f_e and f_s, between the sequences contained in the test set and the ones suggested in both cases. A low value in "RSD f_e" means that the recommended sequences allow the users to essentially "loss" the same amount of time in each view, namely in the recommended sequences we avoid to have complete viewings with some short partial viewings in the middle. Thus the values 23% and 25% indicate, not only a general improvement in function f_e, but also a removal of the short views, that were present in the original test set due to a channel surfing activity.

Similarly, a small value for "RSD f_s" means that the users inside the group are satisfied equally. This parameter is what differentiate the recommendations, without considering the group-type evolution from those that take into account also this aspect. In particular, two values are computed for the test sequences: the first one combines the plain preferences of the users ("%RSD f_s") and is used for evaluating the sequences labeled as "w/o ge", while the second one weights the preference of each single users by considering his/her probability to leave the group in a near future ("%RSD $f_s^\star$") and is used for evaluating the sequences labeled as "with ge". In both cases we can register an improvement (lower value) w.r.t. the test sequences.

TABLE II: 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 both ignoring the possible type evolution ("w/o ge") and considering this prediction ("with ge").

	test set	w/o ge	with ge
%RSD f_e	201%	23%	25%
%RSD f_s	141%	34%	–
%RSD $f_s^\star$	173%	–	27%

The obtained results suggest the effectiveness of the proposed approach and will encourage to further investigate the role of the group-type evolution in the construction of the recommended sequences. In particular, in future work we plan to perform some additional experiments by collecting also the opinion of the users about the provided recommendations in order to better measure the degree of satisfaction, and to investigate how considering the type evolution in the provided sequences can increase the time spent together by a group, namely to prevent that users choose to leave the group during the experience.

VI. Related Work

Recommendation systems represent a well-established research area, and the related literature is therefore vast – see [10] for a survey, and the references therein. Here, we highlight some representative works that 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 [11] 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 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 [12][13]. 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 [14][15]. In case of ephemeral groups, the main problems to solve is to find a suitable measure of the group utility to maximize [16], along with other issues related to the fairness with respect to the different users [17]. Also in these cases, the works do not consider the sequences.

If different activities are related together, e.g., visiting points of interests in a city, listening to songs [18], it is interesting to provide a recommendation for the whole sequence of such activities. The vast majority of the works on this topic [5], [19] (or [20], which takes also into consideration the context) focus on a single user, neglecting the influence of the group composition on the 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 [3] [21] [22]. In [3], the authors provide a system that suggests the path to follow within a museum by a group of visitors. The authors of [21] propose a method for suggesting a sequence of songs to a group of listeners trying to balance the users' satisfaction levels. Herzog [22] considers the Tourist Trip Design Problem (sequence of points of interestes) for a group of users. All the above works share a common limitations: 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 multi objective optimization problem. In addition, we consider how the context influences the recommendation. Our approach is able to improve the quality of recommendation since it is able to represent more complex preferences of the users.

VII. Conclusion

We have presented a system based on a multi-objective optimization algorithm that recommends sequence of activities to dynamic group of users considering their contextual information. The contributions are: (i) the definition of the problem of recommending a sequence of activities, instead of single activity, to groups of users considering a dynamic context represented by the types of users in the group (*adults with kids* or *adults*) and the period during the day (*daytime* or *night*); (ii) the implementation in MapReduce of a multi-objective optimization algorithm (called EMOSA), based on four objective functions: a) the difference between the real experience duration and the desired one, b) the wasting time in each activity, c) the number of activities already done, d) the loss of preferences with respect to the maximum degree of satisfaction; (iii) the evaluation of the proposed solution using a real dataset regarding the watching of TV programs.

The proposed recommendation system is able to propose to a group of users a sequence of entertainments that improves the objective functions w.r.t. the group choices registered in the logs, in particular the dynamic group hypothesis allows to further improve the obtained recommendations. Future work includes the application of the solution to other application domains, like tourist packages generation, and the execution of additional experiments by collecting also the opinion of the users about the provided recommendations.

Acknowledgement

This work was partially supported by the Italian National Group for Scientific Computation (GNCS-INDAM) and by "Progetto di Eccellenza" of the Computer Science Dept., Univ. of Verona, Italy.

References

[1] G. Adomavicius and 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*, vol. 17, no. 6, pp. 734–749, 2005.

[2] M. O'Connor, D. Cosley, J. A. Konstan, and 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.

[3] S. Rossi, F. Barile, C. Galdi, and L. Russo, "Recommendation in museums: paths, sequences, and group satisfaction maximization," *Multimedia Tools and Applications*, vol. 76, no. 24, pp. 26 031–26 055, 2017.

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

[5] M. Quadrana, P. Cremonesi, and D. Jannach, "Sequence-aware recommender systems," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–36, 2018.

[6] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, "Incorporating contextual information in recommender systems using a multidimensional approach," *ACM Transactions on Information Systems*, vol. 23, no. 1, pp. 103–145, 2005.

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

[8] B. Suman, "Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem," *Computers & Chemical Engineering*, vol. 28, no. 9, pp. 1849–1871, 2004.

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

[10] F. Ricci, L. Rokach, and B. Shapira, "Recommender systems: introduction and challenges," in *Recommender systems handbook*. Springer, 2015, pp. 1–34.

[11] X. Ning, C. Desrosiers, and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," in *Recommender systems handbook*. Springer, 2015, pp. 37–76.

[12] A. Jameson and B. Smyth, "Recommendation to groups," in *The adaptive web*. Springer, 2007, pp. 596–627.

[13] J. Masthoff, "Group recommender systems: aggregation, satisfaction and group attributes," in *recommender systems handbook*. Springer, 2015, pp. 743–776.

[14] I. Ronen, I. Guy, E. Kravi, and 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.

[15] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and W. Cao, "Deep modeling of group preferences for group-based recommendation." in *AAAI*, vol. 14, 2014, pp. 1861–1867.

[16] J. Masthoff, "Group modeling: Selecting a sequence of television items to suit a group of viewers," in *Personalized digital television*. Springer, 2004, pp. 93–141.

[17] L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, and M. Shaoping, "Fairness-aware group recommendation with pareto-efficiency," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 107–115.

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

[19] S. Tschiatschek, A. Singla, and A. Krause, "Selecting sequences of items via submodular maximization," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 2667–2673.

[20] R. Srivastava, S. Hingmire, G. K. Palshikar, S. Chaurasia, and 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.

[21] A. Piliponyte, F. Ricci, and J. Koschwitz, "Sequential music recommendations for groups by balancing user satisfaction." in *UMAP workshops*, 2013.

[22] 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.