Pattern Recognition Letters 131 (2020) 341–347

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

An agent-based approach for recommending cultural tours

Flora Amato[a], Francesco Moscato[b,*], Vincenzo Moscato[a], Francesco Pascale[c], Antonio Picariello[a]

[a] Univ. of Naples Federico II, DIETI, Naples, Italy
[b] Univ. of Campania Luigi Vanvitelli, DiSciPol, Caserta, Italy
[c] Univ. of Salerno, DIIN, Fisciano SA, ITaly

ARTICLE INFO

Article history:
Received 17 July 2019
Revised 14 December 2019
Accepted 6 January 2020
Available online 13 January 2020

Keywords:
Multi-agent systems
Recommender systems
Formal planning

ABSTRACT

In the Cultural Heritage domain, the introduction of intelligent systems for planning of cultural visit was very appealing. Indeed when people decide to visit museums or archaeological sites, they usually would like to organize and schedule their time in order to fulfill some requirements and to match, as far as possible, their preferences and needs. In this work, we propose a novel methodology integrating recommendation facilities with agent-based planning techniques in order to implement a planner of routes within cultural sites as museums. In particular, the introduced methodology exploits from one hand a user-centered recommendation strategy to suggest the most suitable cultural items with respect to user's preferences and, from the other one, it leverages multi-agents planning methods to generate the related routes consisting of the sequence of steps necessary to reach precise cultural goals depending on the context, by means of a state space exploration. We present a case of study for the proposed methodology and we describe some experimental results on system efficiency.

1. Introduction

In the age of Big Data *recommender systems* have become essential to provide intelligent browsing of more and more large collections of items, thus supporting users to effectively and efficiently discover "what they need" within the ocean of digital information.

Cultural Heritage undoubtedly represents one of the main application domains that can benefit from the recommendation facilities [10,11,13,19].

In fact, cultural exhibitions are rapidly moving from an old vision where static information are proposed to users, to a novel one leveraging personalized services able to match visitors' personal interests and behaviors. The final goal is to provide a "user-centered information dialog" between a cultural space and its visitors [9].

Generally, cultural digital contents come from distributed and heterogeneous data sources such as digital libraries, archives of cultural foundations, multimedia art collections, web encyclopedias, social media networks (like in [5,15]) and so on.

Indeed when people decide to visit museums or archaeological sites, they usually would like to organize and schedule their time in order to fulfill some requirements, preferences and needs. As an example, they could prefer to see only paintings by a given author or belonging to the same artistic genre, often according to several cultural paths that they have in mind before starting the visit.

Furthermore, sometimes and for particular kinds of museums or archaeological ruins, the planned tours can take very long time to be accomplished and it is not unusual that visitors have to leave before the tour is effectively completed, even if tracking systems are used in order to better manage users' routes [1]. In addition, a wrong scheduling can overload some areas of cultural sites that have to be managed by using proper queuing policies because of safety and security reasons [4,12]. In this last case, a re-plan of visitor routes is necessary.

In this work, we propose a novel methodology integrating recommendation facilities with agent-based planning techniques in order to implement a planner for cultural routes within cultural sites and museums.

A case of study for the proposed methodology is reported and some preliminary experimental results on system efficiency were obtained to validate the approach.

* Corresponding author.
E-mail addresses: flora.amato@unina.it (F. Amato), francesco.moscato@unicampania.it (F. Moscato), vmoscato@unina.it (V. Moscato), francesco.pascale@unisa.it (F. Pascale), picus@unina.it (A. Picariello).

https://doi.org/10.1016/j.patrec.2020.01.005
0167-8655/© 2020 Elsevier B.V. All rights reserved.

2. Related work

2.1. Recommender systems for cultural heritage

As well known, *recommender systems* help people in retrieving information that match their preferences by recommending products or services from a large number of candidates, supporting decisions in various contexts. Just to make some examples, they can suggest what items to buy, which photo or movie to watch, which music to listen, what travels to do, who they can invite to their social network, or even which artwork could be interesting within an art collection [8].

In *Content-Based Filtering* [20], the utility for a user of a given item is estimated exploiting ratings assigned by the same user to other "similar items". Limitations of such techniques are mainly due to the fact that recommendation does not take advantage from information correlated with the behavior of other users. Moreover, the effectiveness of the methods strongly depends on the adopted similarity criterion and *overspecialization* situations (the system can only recommend items that are similar to those already rated by the user) could arise.

Collaborative Filtering [20] is the process of filtering items using the opinions of other people, focusing on the similarity among users. Such approach predicts the utility of items for a given user on the basis of ratings assigned to the same items by "similar users". A drawback of collaborative filtering systems is the *cold start problem*: a recommender is unable to make meaningful recommendations due to an initial lack of ratings.

To overcome the discussed limitations, content-based and collaborative filtering techniques are usually combined within *hybrid* approaches [20].

A recommendation technique is at the basis of an info-mobility system (*Tourist AssIStant*, TAIS) for supporting tourists in a region by means of user actions and preferences analysis. In this system, recommendations depends on the state of the whole environment, and suggestions and plannings about transports to reach cultural assets in the tours are provided [21].

In the last years many works tried to develop new strategies to optimize the planning problem. For example, authors of [16] propose a domain independent approach for heuristics definition based on pattern databases approach. In a similar way, works of [18,24] introduce more efficient search state planners and algorithms. Time-based and other constraints were then introduced in the planning problem by [17] and [6].

Anyway, planning literature usually neglects a common problem: the control of the existence of a plan. Approaches based on state space search are not efficient in detecting if a goal is not reachable. Our methodology is based on a *Multi-Agent models and logic* [22]. The problem of finding a schedule for a cultural tour is managed as a reachability problem: visitors agents express the *goal* of accessing some items, rooms and areas within given deadlines.

3. Recommendation and agent based planning problems

3.1. Recommendation

Recommendation is easily modeled as an information filtering problem. Here, the objective is to reduce and rank the vast amount of multimedia assets from the different data sources that can be associated to a cultural artifact (e.g., images and textual description related to a given picture within a museum), thus simplifying the agent task of searching the most suitable objects for visitors.

Formally, a recommender system deals with a set of *users* $U = \{u_1 \dots, u_m\}$ and a set of *items* $O = \{o_1, \dots, o_n\}$. For each pair (u_i, o_j), a recommender can compute a *rank* $r_{i,j}$ that measures the expected interest of user u_i in item o_j, using a *knowledge base* and a *ranking* algorithm.

The ranking algorithm can exploit different types of information: (i) user preferences and past behaviors, (ii) preferences and behaviors of the whole user community, (iii) items features and how they can match users' preferences, (iv) users feedback, (v) context information and how recommendations can change together with the context.

According to the last generation of recommender systems, the recommendation problem is faced following three distinct information filtering stages [14]:

- In a *pre-filtering* phase, we select for each user u_i a subset $\hat{O}_i \subset O$ containing items that are good candidates to be recommended: such items usually match user preferences and needs.
- During a *ranking* phase, we assign w.r.t. user u_i a rank $r_{i,j}$ to each candidate item o_j in $\hat{O}_i$ using the well-known recommendation techniques (i.e., *content-based, collaborative filtering* and *hybrid* approaches) that can exploit in several ways items features and users' preferences, feedback (in the most of cases in terms of *ratings*) moods (by opinion or sentiment analysis) and behaviors.
- The *post-filtering* steps dynamically excludes, for each user u_i, some items from the recommendation lists; in this way, a new set $\tilde{O}_i \subseteq \hat{O}_i$ is obtained depending on users' feedback and other contextual information (such as data coming from the interactions between users and application/environment). Eventually recommended items can be arranged in *groups* according to additional constraints.

3.2. Agent-based planning

Here, we provide a simplified version of the model we use for our planning problem. The model is based on the multi-agents paradigm. In particular, we consider a variant of *Beliefs, Desires, Intentions* (BDI) logic [23] for our Multi-Agent System model that is represented by the quadruple:

$$(Agents, World, \mathcal{TS}, \mathcal{F})$$

Agents is the set of all agents in the system; *World* represents the environment where agents are executed; $\mathcal{TS}$ is a *Transition System* that resumes possible state transitions of agents in the environment; and, finally, $\mathcal{F}$ is a a set of formulas expressed in first order logic that characterize each state in the World.

We leverage a triple $\langle n, d, v\rangle$ to define variables describing states in the World, where n is the name of a variable, d represents its domain and v is the value assigned to the variable. A *state* $s \in World$ is then a set of variables evaluations. In addition, we call *State Conditions* the set of all formulas in $\mathcal{F}$ that holds in a given state s:

$$StateCondition(s) = \{\phi \in \mathcal{F}, s \in World : s \models \phi\}$$

We note that ϕ can not be a sub-formula of other formulas holding in s.

In this work, we consider a $\mathcal{TS}$ with only one a state condition for each state. If $s \vDash \psi$, $s \vDash \phi$ and $s \vDash \psi \wedge \phi$ then we consider only the last formula as state condition in s.

An Agent is in turn a triple:

$$(Actions, Beliefs, Goals)$$

Actions is a set of possible actions an agent is able to perform. Actions modify the environment changing the World representation.

Beliefs include the knowledge the agents have about the world, the agent itself and other Agents. Notice that an agent may have a belief about the world which in turn is *not* true in the environment: in general, beliefs of agents may not be exact.

Goals is a set of states in $\mathcal{TS}$ that represents the goals an agent want to reach. Since $\mathcal{TS}$ is not available when agents are defined, with an abuse of notation we identify goals with formulas in $\mathcal{F}$ that are satisfied in goal states. We call these formulas *Goal Conditions*. It is easy to note that a goal condition is a state condition for a goal state.

Beliefs are managed as world variables and states (they are practically local world representation in each agent). Agents define the $\mathcal{TS}$ on the world states by means of actions. An action $\alpha \in$ *Actions* is a triple:

$$(name, Precondition, Effects)$$

where *name* is trivially the name of the action; *Effects* is a set of formulas that hold in the new state; *Precondition* is a formula that *evaluates* true in a state s in order to *apply* (execute) the action and to produce a state transition ($s\overrightarrow{\alpha}s'$).

If precondition of α evaluates true in *s, s'* will be the same of *s*, except for variables involved in effects evaluation. The values of these variables have to change in order to satisfy *all* effects in s':

$$\forall\phi \in Effects s' \models \phi$$

In addition we must consider that an agent executing an action can access only to its local representation of the world, i.e. to its beliefs. Hence, if agents' beliefs and world states are not synchronized (i.e., if the agents have wrong beliefs about the world), it is possible that precondition is evaluated true on beliefs, but *not* on world state.

In order to apply an action, we must execute the following two steps:

1. an agent tries to apply effects in a state s producing a transition from s to s' if precondition is evaluated true on its *beliefs*;
2. if precondition evaluates true in the world too, then $s\overrightarrow{\alpha}s'$ both in agent's Beliefs and in the world too.

$\mathcal{TS}$ will be then the transition system defined by the application of all actions in any state of the world, performed by *all* agents in the model. The execution of an action to build $\mathcal{TS}$ must follow the two steps previously defined. State transitions apply both to agents beliefs and to the world. Anyway precondition control is enacted on beliefs first. If evaluation fails on beliefs, the action is not applied even if precondition would evaluate true on the world.

In this model, a *Plan* to reach a goal $\mathcal{G}$ with given goal condition is a *path* from a starting state to a state where the Goal Condition holds.

In a Multi Agent System, actions in a transition system can be executed by different agents, even concurrently. We consider here a path as a linear scheduling of concurrent applications of actions.

A Planning problem hence, is the problem of finding such a path or to state that the requested goal is unreachable in the current environment.

4. Methodology and framework

4.1. The recommendation strategy

The basic idea behind the adopted strategy is that when a user is browsing a multimedia art collection in a real o virtual environment, the recommender system: (i) selects a set of useful *candidate* items on the base of user actual needs and preferences (*pre-filtering stage*); (ii) opportunely assigns to these items a rank, previously computed exploiting items' intrinsic features, users' past behaviors, and also leveraging users' opinions and feedback (*ranking stage*); (iii) dynamically, when a user "selects" as interesting one or more of the candidate items.

In our approach, items to be recommended are multimedia data (i.e. texts, images, videos, audios) related to specific artworks (e.g. paintings, sculptures or other kinds of artifacts that can be exhibited within a museum). As in the majority of multimedia systems, items are described at two different levels: (i) from an "high-level" perspective, by one or more set of *metadata* that do not depend on the type of multimedia object[1]; (ii) from a "low-level" perspective, by a set of *features* that are different for each kind of multimedia object.

To cope with the heterogeneity of multimedia data, our recommender system is constituted by a *multichannel browser* [2].

Each single channel is then specialized to recommend multimedia items of a given type. Recommended items of different types can be eventually grouped in a unique "multimedia presentation". We describe the recommendation strategy provided by each single browsing channel.

4.1.1. Pre-filtering stage using user preferences

In the *pre-filtering* stage, our aim is to select for a given user u_h a subset $\hat{O}_h \subset O$ containing items that are good "candidates" to be recommended: such items usually have to match some (static) user preferences and (dynamic) needs from the "high-level" perspective.

Each item subjected to recommendation may be represented in different and heterogeneous feature spaces. For instance, a photo may be described by a set of metadata as title, set of tags, description, etc. The first step consists in clustering together "similar" items, where the similarity should consider all (or subsets of) the different spaces of features.

To this goal, we employ *high-order star-structured co-clustering* techniques - that some of the authors have adopted in previous work [7] - to address the problem of heterogeneous data filtering, where a user is represented as a set of vectors in the same feature spaces describing the items.

4.1.2. Ranking stage using user behavior and items similarity

The main goal of this stage is to automatically rank the set of items O embedding in a collaborative learning context.

(user preferences are represented modeling the choice process in recommender system considering users' *browsing behaviors*) their *intrinsic features* (those on the top of which it is possible to introduce a *similarity* notion). In particular, we use a novel technique that some of the authors have proposed in previous works [3].

Our basic idea is to assume that when an item o_i is chosen after an item o_j in the same user *browsing session* (and both the explored items have been positively rated or have captured attention of users for an adequate time), this event means that o_i "is voting" for o_j. Similarly, the fact that an item o_i is "very similar" in terms of some intrinsic features to o_j can also be interpreted as o_j "recommending" o_i (and viceversa).

Thus, we are able to model a browsing system for the set of items O as a labeled graph (G, l), where: (i) $G = (O, E)$ is a *directed graph*; (ii) $l : E \rightarrow \{pattern, sim\} \times R^+$ is a *labeling function* that associates each edge in $E \subseteq O \times O$ with a pair (t, w), where t is the type of the edge which can assume two enumerative values (*pattern* and *similarity*) and w is the weight of the edge.We list two different cases:

1. a *pattern label* for an edge (o_j, o_i) denotes the fact that an item o_i was chosen immediately after an item o_j and, in this

[1] In the Cultural Heritage domain different harvesting sets of metadata and possibly domain taxonomies or ontologies can be considered.

case, the weight w^i_j is the number of times o_i was chosen immediately after o_j;

2. a *similarity label* for an edge (o_j, o_i) denotes the fact that an item o_i is similar to o_j and, in this case, the weight w^i_j is the "similarity" between the two items. Thus, a link from o_j to o_i indicates that part of the importance of o_j is transferred to o_i.

Given an item $o_i \in O$, its *recommendation grade* $\rho(o_i)$ is defined as follows:

$$\rho(o_i) = \sum_{o_j \in P_G(o_i)} \hat{w}_{ij} \cdot \rho(o_j) \tag{1}$$

where $P_G(o_i) = \{o_j \in O, (o_j, o_i) \in E\}$ is the set of predecessors of o_i in G, and $\hat{w}_{ij}$ is the normalized weight of the edge from o_j to o_i.

We note that for each $o_j \in O$ $\sum_{o_i \in S_G(o_j)} \hat{w}_{ij} = 1$ must hold, where $S_G(o_j) = \{o_i \in O | (o_j, o_i) \in E\}$ is the set of successors of o_j in G.

In [3], it has been shown that the ranking vector $R = [\rho(o_1) \ldots \rho(o_n)]^T$ of all the items can be computed as the solution to the equation $R = C \cdot R$, where $C = \{\hat{w}_{ij}\}$ is an ad-hoc matrix that defines how the importance of each item is transferred to other items.

The matrix can be seen as a linear combination of:

- a *local browsing matrix* $A_h = \{a^h_{ij}\}$ for each user u_h, where its generic element a^l_{ij} is defined as the ratio of the number of times item o_i has been chosen by user u_h immediately after o_j to the number of times any item in O has been chosen by u_h immediately after o_j;
- a *global browsing matrix* $A = \{a_{ij}\}$, where its generic element a_{ij} is defined as the ratio of the number of times item o_i has been chosen by any user immediately after o_j to the number of times any item in O has been chosen immediately after o_j;
- a *similarity matrix* $B = \{b_{ij}\}$ such that b_{ij} denotes the similarity between two items o_i and o_j (a semantic relatedness [3] based on a set of taxonomies using some high-level features values [2], eventually combined with a low-level features comparison using Windsurf multimedia libraries.

The successive step is to compute *customized* rankings for each individual user. In this case, we can rewrite previous equation considering the ranking for each user as $R_h = C \cdot R_h$, where R_h is the vector of preference grades, customized for a user u_h considering only items in the related $\hat{O}_h$.

We note that solving the discussed equation corresponds to finding the stationary vector of C, i.e., the eigenvector with eigenvalue equal to 1. In [3], it has been demonstrated that C, under certain assumptions and transformations, is a real square matrix having positive elements, with a unique largest real eigenvalue and the corresponding eigenvector has strictly positive components. In such conditions, the equation can be solved using the *Power Method* algorithm.

Rank can be finally refined using user attached sentiments and ratings (see [14] for more details).

4.1.3. Post-filtering stage using context information

We have introduced a *post-filtering* method for generating the final set of "real" candidates for recommendation using *context* information. The context is represented by means of the well-known *key-value* model using as dimensions some of the different feature spaces related to items.

In our system, context features can be expressed either directly using some *target items* (e.g. objects that have positively captured user attention) or specifying the related values in the shape of *constraints* that recommended items have to satisfy.

Assume that a user u_h is currently interested in a target item o_j. We can define the set of candidate recommendations as follows:

$$\tilde{O}_{h,j} = \bigcup_{k=1}^{M} \{o_i \in \hat{O}_h \mid a^k_{ij} > 0\} \cup \{o_i \in NNQ(o_j, \hat{O}_h)\} \tag{2}$$

The set of candidates includes the items that have been accessed by at least one user within k steps from o_j, with k between 1 and M, and the items that are most similar to o_j according to the results of a *Nearest Neighbor Query* ($NNQ(o_j, \hat{O}_h)$) functionality.[2] The ranked list of recommendations is then generated by ranking the items in $\tilde{O}_{h,j}$, for each item o_j selected as interesting by user u_h, using the ranking vector R_h thus obtaining the final set $\tilde{O}_h$.

Finally, for each user all the items that do not respect possible context constraints are removed from the final list.

4.2. The visit planning

For Planning mode, the environment (*World*) is modeled in terms of states defined by means of ⟨*n, d, v*⟩ triples.

In addition, Agents are modeled in terms of Beliefs and Actions. Actions requires the definition of (*name, Precondition, Effects*) triples, but we introduce in this phase an extension of the model since we need some additional information during planning. Hence we extend Actions definition to the quadruple:

$$(name, Precondition, Effects, VarInfo)$$

where *VarInfo* set is necessary to specify variable domains, quantification(universal or existential), and eventually other properties.

A planner engine produces a scheduling of actions representing the plan to achieve the requested goal. Then Agents perform plans action by action at run time. The Plan search algorithm in the Counter Example Planner works in four steps:

(1) The TA (Timed Automata) translator implements an algorithm for Agent models translation from the model in previous section to Timed Automata formalism; (2) Transitions in TA are weighted and filtered by using the methodology introduced in Section 4.1- (3) The TA is passed to the UPPAAL Reasoning Engine in order to produce a counter example (if any) to the following formula: **"A [] !'GoalCondition'"** that is *"is it true that from initial state, only states where the goal condition is not satisfied are reachable?"*; (4) If the previous property is satisfied, then no plan exists for the current goal; otherwise, UPPAAL returns a *counter example* that is a path from the initial state to a state where the goal condition is satisfied, in form of an *Uppaal trace*; (5) The Uppaal trace translator perform the translation from trace to a sequence of actions that is the agent plan.

5. Case studies

The case study we propose, consists in a scenario of a museum, populated by a group of tourists, who want to visit objects of interest inside it. Tourists have their own personal goals. We can have tourists interested in visiting the entire museum, or only particular areas (e.g. Ancient History or Renaissance).

The Environment (*World*) provides the description of all the areas of the museum and information regarding the entry and exits points of each museum room. It is assumed that the museum is properly equipped with a system of sensors, capable of understanding the environmental conditions of the room and the positions of tourists.

Wireless sensors near items and doors in the area provide information about position of visitors, their number in areas etc.

[2] Note that a positive element a^k_{ij} of A^k indicates that o_i was accessed exactly k steps after o_j at least once.

Identifier	Explanations
$item_i.visitors$;	the number of visitor seeing an item;
$item_i.weight$	the *importance* of the item in the museum;
$item_i.maxVisitors$	the maximum number of visitors allowed for an item at the same time;
$item_i.meanVisitors$	the mean number of visitors on an item at the same time;
$item_i.meanTimetoVisit$	the mean time visitors stay near the item;
$item_i.queueLimit$	the maximum number of persons in queue for an item;
$item_i.visitorsInQueue$	the number of persons in queue for an item;
$item_i.queueLimit$	the maximum number of persons for a queue;
$item_i.meanTimeinQueue$	the mean time in queue for an item;

Fig. 1. Beliefs regarding the states of tourists near an Object of Interest.

Identifier	Explanations
isAdjacent($item_i$,$World.item_j$)	a facts defining the adjacency of items and areas in the Museum
inArea($Area_i, World.item_k$)	a fact defining if an item is in $Area_i$ in the Museum
isFull($Area_i$)	when an Area is full and cannot accept more visitors
isQueueFull($Area_i$)	returns true if the queue to access to the area is Full
isInArea($Visitor_i, Area_i$)	returns true if a visitor is inside an area
canMove($Visitor_i, Area_i, Area_j$)	states if a visitor can move directly from an area to another
isOnItem ($Visitor_i, item_j$)	defines if a Visitor in near an Item
isOnQueueforArea($Visitor_i, Area_j$)	defines if a Visitor is queued to enter into and are
isOnQueueforItem($Visitor_i, item_j$)	defines if a Visitor is queued for an Item

Fig. 2. Beliefs regarding the environment.

Identifier	Explanations
$Visitor_i.time$	the time spent by visitor at the museum
$Visitor_i.visited(item)$	is true if visitor saw the item
$Visitor_i.numberInGroup$	stores the number of persons in a group of visitors
$Visitor_i.position$, $Visitor_i.previousPosition$	store the positions of a visitor
$Visitor_i.atItem$	maintains the id of the Item which visitor is currently seeing
$Visitor_i.meanTimetoSee$	is the mean time a visitor takes to see and item
$Visitor_i.visitDeadline$	is the deadline (if any) to visit the whole museum
$Visitor_i.areaDeadline(Area_i)$	is the deadlines (if any) to visit each area of the museum
$Visitor_i.itemDeadline(Item_i)$	is the deadlines (if any) to visit each item of the museum
$Visitor_i.previousRoute$ and $Visitor_i.nextRoute$	are the path the visitor has already walked and the scheduled route respectively

Fig. 3. Beliefs characterizing Visitors.

Identifier	Explanations
Move(from,to	from an area to another or from an item to another
See(item)	an item in an area
Leave(item)	an item in an Area
Enqueue(Area) or (item)	for an area or an Item
Exit(Area) or (Item)	from a queue

Fig. 4. Actions of proactive agents.

It is also assumed that the museum is populated by *N* groups of *K* tourists, and that the museum exposes *M* art pieces.

We use Agent as key point of the museum model. *Resource Agents* include Areas and Items because they are not characterized by reactive or active behaviors; *Proactive agents* include visitors and groups, because they can assume reactive or active behaviors.

We model everything in the museum as an agent: Areas and Items are considered as *Resource Agents* since they have no reactive or active behaviors; visitors (and group of visitors) are modeled as Proactive agents. *Resource Agents Beliefs* (Fig. 1) include information about the belief on Areas and Items. They comprise the position and the name of the Area, the name of the author of the art piece, information about the date in which the item was shaped, the school the item and the authors belong etc.

We use a dot notation to indicate beliefs, for example we use a notation like *Item$_i$.author* or *Item$_i$.date* for addressing the authors of a item or a date in which the item was released.

We defined belief for stating the maximum number of tourist that can stopover in the proximity of an art piece, and the number of tourists that can stay in queue to visit an art piece. In similar way, we defined beliefs stating information about the area. The environment is modeled by formulas reported in Fig. 2. Fig. 3 reports the main Beliefs characterizing Visitors.

Main Actions that proactive agents are able to execute are modeled in Fig. 4.

To report a concrete case of use, the first action can be codified with the statements reported in are reported in Fig. 5, where *Sensor* is the name of the domain defining all possible sensors in the museum.

Identifier	Explanations
Precondition:	(position==from) AND (to!=from) AND (isAdjacent(from,to)==true) AND (isFull(to)==false) AND $visitor.time < visitor.visitDeadline$
Effects:	position=to, prevposition=from, from.visitors=from.visitors-1; to.visitors=to.visitors+1
Variables:	(from,"Sensor","Existential"), (to,"Sensor","Existential")

Fig. 5. Model of first action executed by proactive agents.

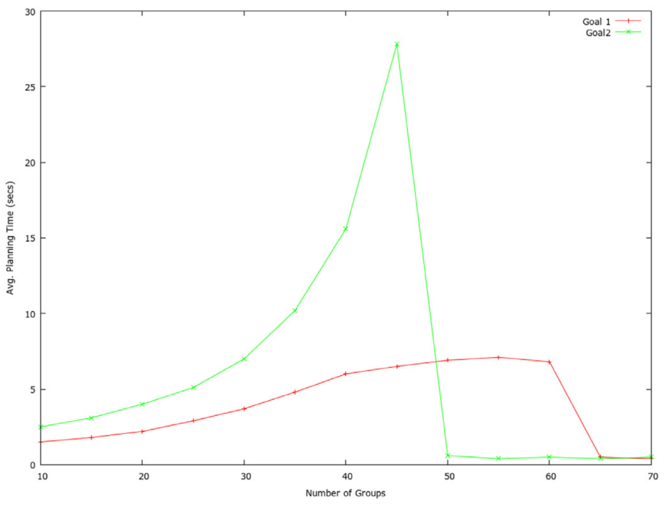

Fig. 6. Re - Planner average planning time.

Goals we want to reach can be different:

$G_1 : \{\forall item Visitor.visited(item) == true\}$

i.e. the goal of visiting all items in the museum (under global deadline) or:

$G_2 : \{max \sum_i (item_i.weight * item_i.visited)\}$

i.e. the goal of visiting major items in the museum within deadline, etc.

In order to evaluate the performance of the planner, we perform a series of simulation on different tourists group, made of different number of people. In general, a single group is made of a fixed number of people: in our experiments, a single group is made of 15 visitors. We consider a single visitor and analyse its route. It is influenced to a possible deadline expiration in visiting the area of the museum, given by a number of maximum *World* configuration. In our example, to visit 500 items, placed in 50 areas, 10 items for area, the maximum number of visitor in an area can be of 25 persons. In addition, we simulated the replan options for both goals. The replan is necessary in the case of *World* configuration, if a tourist cannot reach his goal before its deadline expires. Fig. 6 depicts the average planning time for two goals.

6. Conclusions and future works

In the domain of cultural heritage, the introduction of a system for visits planning is very appealing. When people go to visit museums or archaeological sites, they usually want to schedule their time in order to fulfill some requirements, like they would see first paintings from the same period or region. The introduction of a tool able to adapt tours to user expectation and time requirements can noticeably improve users' experiences and satisfaction. Furthermore, sometimes, tours can take long time to finish and it is not unusual that visitors run short of time and they have to leave before the end of the tour, visitors have to leave the museum before the tour is completed. In addition, wrong scheduling can overload some areas, that have to be managed by using proper queuing policies, also because of safety and security. This work applies a methodology able to mix classical and novel planning technique in order to implement a planner for routes of visitors in museums. The methodology uses techniques for classical planning to find a sequence steps to reach a goal by state space exploration and define heuristics in order to improve efficiency. A case of study aiming to describe the application of the methodology has been presented in order to validate the methodology.

Declaration of Competing Interest

None.

CRediT authorship contribution statement

Flora Amato: Conceptualization, Data curation, Formal analysis, Writing - original draft, Writing - review & editing. **Francesco Moscato:** Conceptualization, Data curation, Formal analysis, Writing - original draft, Writing - review & editing. **Vincenzo Moscato:** Conceptualization, Data curation, Formal analysis, Writing - original draft, Writing - review & editing. **Francesco Pascale:** Conceptualization, Data curation, Formal analysis, Writing - original draft, Writing - review & editing. **Antonio Picariello:** Conceptualization, Data curation, Formal analysis, Writing - original draft, Writing - review & editing.

References

[1] I. Ahmed, A. Ahmad, F. Picciallі, A. Sangaiah, G. Jeon, A robust features-based person tracker for overhead views in industrial environment, IEEE Internet Things J. 5(3), 1598–1605.

[2] M. Albanese, A. Chianese, A. d'Acierno, V. Moscato, A. Picariello, A multimedia recommender integrating object features and user behavior, Multimed. Tools Appl. 50 (3) (2010) 563–585.

[3] M. Albanese, A. d'Acierno, V. Moscato, F. Persia, A. Picariello, A multimedia recommender system, ACM Trans. Internet Technol. 13 (1) (2013) 3.

[4] F. Amato, A. Chianese, V. Moscato, A. Picariello, G. Sperli, Snops: a smart environment for cultural heritage applications, in: Proceedings of the Twelfth International Workshop on Web Information and Data Management, ACM, 2012, pp. 49–56.

[5] F. Amato, V. Moscato, A. Picariello, F. Piccialli, G. Sperl, Centrality in heterogeneous social networks for Lurkers detection: an approach based on hypergraphs, Concurr. Comput. 30(3).

[6] J.A. Baier, F. Bacchus, S.A. McIlraith, A heuristic search approach to planning with temporally extended preferences, Artif. Intell. 173 (5) (2009) 593–618.

[7] I. Bartolini, V. Moscato, R.G. Pensa, A. Penta, A. Picariello, C. Sansone, M.L. Sapino, Recommending multimedia visiting paths in cultural heritage applications, Multimed. Tools Appl. 75 (7) (2016) 3813–3842.

[8] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl.-Based Syst. 46 (2013) 109–132.

[9] J.P. Bowen, S. Filippini-Fantoni, Personalization and the web from a museum perspective, Museums and the Web, 4, Archives & Museum Informatics, Toronto, Canada, 2004.

[10] A. Castiglione, F. Colace, V. Moscato, F. Palmieri, Chis: a big data infrastructure to manage digital cultural items, Future Gener. Comput. Syst. 86 (2018) 1134–1145.

[11] A. Chianese, F. Marulli, F. Piccialli, Cultural heritage and social pulse: a semantic approach for CH sensitivity discovery in social media data, pp. 459–464. doi:10.1109/ICSC.2016.50.

[12] A. Chianese, F. Piccialli, I. Valente, Smart environments and cultural heritage: a novel approach to create intelligent cultural spaces, J. Locat. Based Serv. 9 (3) (2015) 209–234.

[13] F. Clarizia, F. Colace, M. De Santo, M. Lombardi, F. Pascale, A sentiment analysis approach for evaluation of events in field of cultural heritage, in: 2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS), 2018, pp. 120–127.

[14] F. Colace, M. De Santo, L. Greco, V. Moscato, A. Picariello, A collaborative user-centered framework for recommending items in online social networks, Comput. Hum. Behav. 51 (2015) 694–704.

[15] F. Colace, L. Greco, S. Lemma, M. Lombardi, F. Amato, V. Moscato, A. Picariello, Contextual aware computing and tourism: a case study, in: 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), IEEE, 2015, pp. 804–808.
[16] J.C. Culberson, J. Schaeffer, Pattern databases, Comput. Intell. 14 (3) (1998) 318–334.
[17] P. Haslum, H. Geffner, Heuristic planning with time and resources, in: Sixth European Conference on Planning, 2014.
[18] J. Hoffmann, B. Nebel, The ff planning system: fast plan generation through heuristic search, J. Artif. Intell. Res. (2001) 253–302.
[19] V. Moscato, Big data in cultural heritage, Encyclopedia of Big Data Technologies, 2019, doi:10.1007/978-3-319-63962-8_29-1.
[20] F. Ricci, L. Rokach, B. Shapira, Recommender systems: introduction and challenges, in: Recommender Systems Handbook, Springer, 2015, pp. 1–34.
[21] A.V. Smirnov, A.M. Kashevnik, A. Ponomarev, Context-based infomobility system for cultural heritage recommendation: Tourist assistant–tais, Pers. Ubiquitous Comput. 21 (2) (2017) 297–311.
[22] I. Trencansky, R. Cervenka, Agent modeling language (AML): a comprehensive approach to modeling mas, Whitestein Ser. Softw. Agent Technol. Auton. Comput. 29 (2005) 391–400.
[23] M. Wooldridge, Agent-based software engineering, in: IEE Proceedings on Software Engineering, 1997, pp. 26–37.
[24] R. Zhou, E.A. Hansen, Breadth-first heuristic search, Artif. Intell. 170 (4) (2006) 385–408.