Entertainment Computing 28 (2018) 32–48

Contents lists available at ScienceDirect

Entertainment Computing

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

Player behavior and personality modeling for interactive storytelling in games

Edirlei Soares de Lima[a,*], Bruno Feijó[b], Antonio L. Furtado[b]

[a] Rio de Janeiro State University (UERJ), Department of Computational Modeling, Nova Friburgo, RJ, Brazil
[b] Pontifical Catholic University of Rio de Janeiro (PUC-RIO), Department of Informatics, Rio de Janeiro, RJ, Brazil

ARTICLE INFO

Keywords:
Player modeling
Behavior
Personality
Quest generation
Games
Interactive storytelling

ABSTRACT

Current video games use simple methods to deal with interactive narratives and the enormous variety of player types. In this paper, we propose a novel approach to interactive storytelling in games, in which the quests and the ongoing story are determined in view of individual personality traits and behavioral attitudes in a non-deterministic way. Our method starts the process employing a new technique to assess the player's personality traits according to the well-known Big Five model. These traits are then used by a nondeterministic planning algorithm to define adaptive goal hierarchies. In addition, an artificial neural network is trained to predict player behaviors in real-time, allowing partial-order planning operators to use player behaviors and personality traits as logical terms in their preconditions. With this approach, a richer individualized experience is provided to the player, while preserving consistency with the conventions of the chosen genre.

1. Introduction

Video games add new dimensions to traditional storytelling by allowing players to change narratives through their own actions. In modern Role-Playing Games (RPGs), this is usually done by adopting branching storylines based on key choices presented to players at certain points of the game. Some RPGs, such as *Mass Effect 2* (BioWare, 2010), *Dragon Age: Inquisition* (BioWare, 2014), and *The Witcher 3: Wild Hunt* (CD Projekt RED, 2015), perform this so well that they provide the player with a real sense of control over the story. However, such branching points are usually presented through specific dialog choices or predetermined actions (e.g. killing or forgiving an enemy, collecting or not collecting a specific item), which reduces the player's sense of agency (i.e. the impression of controlling his/her own actions).

Currently, both game industry and game consumers have great interest in new forms of interactive storytelling that may provide games with truly interactive stories, in which all in-game player's actions and preferences can affect the development of the narrative. However, understanding player preferences and interpreting in-game behaviors in real-time is not an easy task. This problem involves an active topic of research on artificial intelligence, known as player modeling.

Player modeling is the study and use of artificial and computational intelligence techniques [1] for the construction of computational models of players, which includes cognitive, affective and behavioral characteristics. In general, a player model is an abstract description of the player's characteristics in the real world or in the game environment [2]. Indeed, the construction of effective player models involves a multidisciplinary intersection of the fields of affective computing, experimental psychology, human-computer interaction, big data, and analytics, which are part of the so called "game analytics" [3].

Nowadays, analyzing game data is a common practice widely employed in the game industry to validate level design or improve the player experience [3,4]. Many commercial games are known for adopting player modeling techniques, such as *Silent Hill: Shattered Memories* (Konami, 2009), which dynamically creates personality models of the players and uses them to adapt gameplay elements [5]; *League of Legends* (Riot Games, 2009), which explores the analysis of gameplay data to design new content updates [6]; and *Left 4 Dead* (Valve, 2008), which uses player modeling to adapt the difficulty of the game's challenges in response to the player's actions. Even though player modeling has been successfully applied to commercial games and widely explored by academic researchers, only few works treat the prediction of actual player behavior or the use of this information to create interactive game narratives.

We consider actual player behavior to be the way in which the player acts or conducts him or herself in the game. For instance, the player may behave aggressively, impulsively, or cautiously when facing dangerous situations. Behavior, in general, is not only complex, but also

* Corresponding author.
E-mail address: edirlei.lima@uerj.br (E.S. de Lima).

https://doi.org/10.1016/j.entcom.2018.08.003
Received 2 March 2018; Received in revised form 21 August 2018; Accepted 31 August 2018
Available online 01 September 2018
1875-9521/

dynamic. A behavioral description for one occasion is likely to be invalid for another occasion. In fact, behavior is susceptible to variation in consequence of any change in time, place, emotion, and social context [7]. In addition, a player's behavior within a game environment may be very different from the same person's behavior when dealing with real world situations.

Although understanding player behavior can help games to adapt their narratives as an indirect consequence of the players' actions, there is no guarantee that they will enjoy the resulting narratives. An important factor to be considered when adapting game narratives is the personality of the players, which is known to exert a major influence on their preferences and expectations for future narrative events [8]. When a game is aimed at providing pleasurable entertainment, having some information about the current player's preferences is vital to create satisfying playing experiences.

In this paper, we propose a novel approach to interactive storytelling in games based on player behavior and personality modeling, hierarchical task decomposition and nondeterministic planning.[1] The proposed method can generate dynamic and nondeterministic hierarchical quests[2] that are directly or indirectly affected by the player's *personality* and *in-game behavior*, which are both modeled in terms of the Big Five factors [9]. These factors are typically used to describe individual personality, but they can also be used to explain human behaviors [40,41].

Before the game starts, our method uses a new technique to find the player's *personality traits* based on the Big Five model. These traits are used by the nondeterministic planning algorithm to define goal hierarchies during gameplay. Also we define behavioral aspects based on Big Five factors, which are associated with general in-game *player behaviors*. In our method we use an artificial neural network to predict behavioral aspects based on 32 statistical features extracted from the gameplay. Once the neural network is trained, the system can predict player behaviors at any time. After the game starts, *partial-order planning operators* take the following characteristics as terms of its *preconditions*: player behaviors (which are dynamic and time dependent) and personality traits (which are "persisting" characteristics that are consistently demonstrated despite changing circumstances or game environment).

The above-mentioned new technique to determine the player's personality traits uses a preliminary short session with story-related scenes, which are based on the 10-item Big Five inventory called BFI-10 [48]. For the sake of concisely presenting our method, we name this technique as *Big Five Game Inventory* (BFGI-10).

In this paper, we evaluate the following elements of our approach to interactive storytelling in games: the behavioral model (by assessing the accuracy and performance of the Neural Network); the personality model (by comparing the results of our Big Five Game Inventory and BFI-10); and the hierarchical quest generation algorithm (by analyzing its scalability and performance). Since in our project we take for granted the desirability of player modeling features in games, provided that they are well-calibrated and, in addition, do not raise privacy concerns, we did not propose to compare the users' reactions to game versions with and without such features.

The paper is organized as follows. Section 2 reviews related work. Section 3 gives an overview of the system architecture and introduces the testbed game used to validate our method. Section 4 presents the proposed player behavior model. Section 5 describes the proposed personality model. Section 6 presents the proposed method for quest generation based on nondeterministic planning and player modeling. Section 7 contains an evaluation of our method. Section 8 offers concluding remarks.

2. Related work

There are several works on player modeling in the literature, which started with the concept of player types. Indeed, one of the earliest attempts to create player models came in 1996 when Richard Bartle [12] proposed his four player types (Achievers, Socializers, Explorers, and Killers). Following Bartle's work, Bateman and Boon [13] created another model using the Myers-Briggs personality indicator [14] to categorized players into four classes: Conqueror, Manager, Wanderer and Participant. Yee [15] empirically grounded Bartle's original model and found that player motivation has three main components: achievement, social, and immersion. Moreover, taking inspiration from neurobiological research, Nacke et al. [16] proposed seven player archetypes (Seeker, Survivor, Daredevil, Mastermind, Conqueror, Socializer, Achiever). However, as previously pointed by Tuunanen and Hamari [17], type-based approaches are very limited, because types provide only a superficial information about the player, which can be even more blurred considering that most players cannot be adequately categorized into a single group.

In recent years there have been several successful implementations of player modeling in games, whose applications include the use of player models for adapting player experience, game balancing, personalized content generation, playtesting analysis and game authoring. Missura and Gärtner [18] explore the use of clustering and classification techniques to dynamically adjust the difficulty of a shooter game. Their method uses k-means and support vector machines to classify players into different types based on gameplay data. Weber and Mateas [19] employ a series of classification algorithms for recognizing player strategies in *StarCraft* (Blizzard Entertainment, 1998). Mahlman et al. [20] use several supervised machine learning algorithms, trained with a set of player behavior data extracted from the game *Tomb Raider: Underworld* (Crystal Dynamics, 2008), in order to predict when a player will stop playing the game and, if the player completes the game, how long will it take to do so. Machado et al. [21] and Spronck and den Teuling [22] explore player modeling in the context of the game *Civilization IV* (Firaxis Games, 2005). Machado et al. [op. cit.] create models of virtual agent's preferences using classifiers based on support vector machines, and Spronck and den Teuling [op. cit.] use a sequential minimal optimization (SMO) classifier to build a player model to predict specific preference values. In a recent work, Valls-Vargas et al. [4] propose a player modeling framework to capture and predict play style using episodic segmentation of gameplay traces and sequential machine learning techniques. Their framework utilizes multiple models that include predictions from previous time intervals to identify how players change play style over time.

The use of player modeling has also been explored in interactive storytelling systems. Barber and Kudenko [23] present an interactive story generator system that learns the personality of its users by applying predefined increments or decrements to a vector of personality traits, such as honesty and selfishness, in response to the users' decisions. Seif El-Nasr [24] presents an interactive storytelling system called Mirage, where both player behavior and personality are modeled in order to allow users to participate in a more engaging drama. The system tracks user's actions to adjust a vector of values representing tendencies toward character traits (heroism, violence, self-interestedness, and cowardice). Sharma et al. [25] present an interactive storytelling system that combines past captured game traces and player survey data to create player models, which are used to dynamically determine the next plot point that is best suited to specific users. Thue et al. [26] present PaSSAGE, an interactive storytelling system that uses player modeling to automatically learn a model of the player's preferences through observations of the player in the virtual world, and then uses the model to dynamically select the content of an interactive

[1] We use the term "nondeterministic planning" for planning problems in which the planning domain is a nondeterministic state-transition system, i.e. an action may have more than one possible outcome [10],[11].

[2] Quests are missions or objectives to be accomplished by *avatars* (which are game characters controlled by human players).

story. The player is modeled by a vector, where each dimension is the strength of one of the Laws' stereotypes [27]. As the player performs actions, dimensions are increased or decreased in accordance to predefined rules. Ramirez and Bulitko [28] use this player model with a reward function in such a way that, when several narratives are generated, the one that maximizes this function is automatically selected.

The Big Five model was used in some previous works on player modeling. Van Lankveld et al. [29] investigate whether a personality profile can be determined by observing the player's behavior in a customized scenario for the game *Neverwinter Nights* (Bioware, 2002). They adopted the Big Five model to define the player's personality profile. In a recent work, Nagle et al. [5] explore the application of the Big Five model for difficulty adjustment in a first-person shooter game. They present a linear regression model to predict difficulty adaptations that maximize enjoyment and gameplay duration based on player personality.

The Big Five model has also been used to investigate personality elements among gamers, and their choice of character and style of play in MMORPGs [30]. Furthermore, several studies have examined the relationship between the Big Five traits and problematic gaming [31]. More general approaches to produce a generic measure of player style that goes beyond RPG variants have been proposed [32,33], which are based on an alternative personality model to Big Five, named HEXACO [34]. This name is an acronym for Honesty–Humility, Emotionality, Extraversion, Agreeableness, Conscientiousness, and Openness to experience. However, the goal of all these works is to understand the preferences, disorders and abilities of the players – which is not within the scope of the present paper.

Even though player modeling has been widely explored in games, little work has been done to use the player's behavior to adapt game narratives. The few previous works reviewed above that explore the Big Five model for player modeling use it only to establish personality profiles. In addition, most previous works on behavior modeling are based on very simplified models of player archetypes, which fail in representing blended behaviors, as well as in providing more detailed information about the actual player behavior.

3. The planning system

3.1. Hierarchical quests

The structure of the game's narrative (Fig. 1) is represented as a quest hierarchy, where the entire game can be described as a single quest, which can in turn be decomposed into lower level sub-quests and/or events (shaded rectangles in Fig. 1). We use the term *event* to indicate a *primitive* action to be executed by the game controller or imposed by the player. In Fig. 1, the dotted lines indicate *decomposition* and the arrows indicate *sequential direction*. When the story is completed a totally ordered sequence of events ($Event_1$ to $Event_7$ in the example of Fig. 1) is determined.

Quests are logically modeled to have multiple goal states, so they can be completed in different ways depending on the player's actions. Consequently, the results of sub-quests can influence the progression of their parent quests and dynamically change the entire storyline of the game. This dynamism is achieved by modeling each quest as a non-deterministic planning problem.

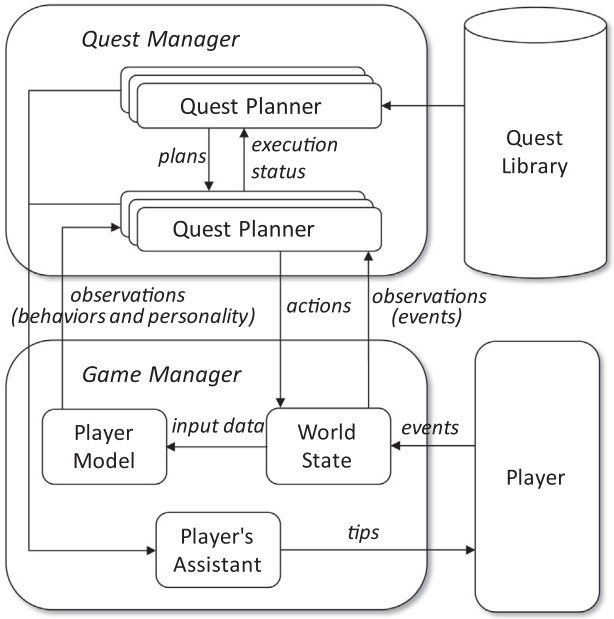

Fig. 2. Architecture of the quest generator system.

3.2. System architecture

The architecture of the system under description in this paper was built as an extension of our previous work on hierarchical quest

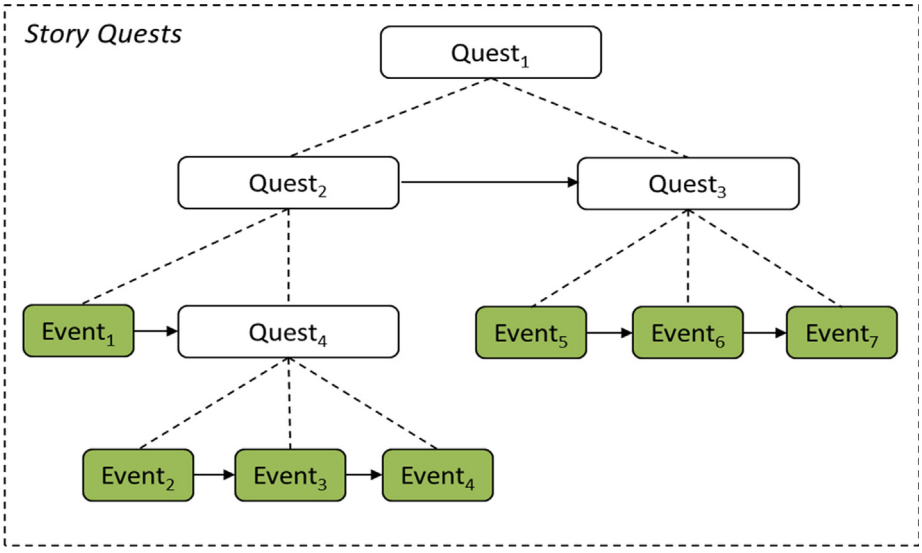

Fig. 1. Hierarchy of quests.

generation [35]. Fig. 2 illustrates the architecture of our system, which is consistent with the conceptual model for dynamic planning presented by Ghallab et al. [10]. In our implementation, the Quest Manager is responsible for controlling multiple instances of planners and plan monitors. The Quest Planner is responsible for generating a logical plan of actions to achieve an authorial goal of the quest starting from the current state of the world. The Quest Monitor is responsible for monitoring the execution of the plan to verify the occurrence of changes introduced by the player. The Game Manager manages the game world by updating the current World State according to the *events* produced by actions performed by the Player during the gameplay. In addition, the Game Manager maintains the Player Model, which is updated with *input data* extracted from the World State. While the World State aggregates all information about the events that occur in the game as result of direct player actions, the Player Model maintains a more general description of the player personality and the actual in-game player behavior. The observations (*events*, *behaviors*, and *personality*) produced by the Player Model and the World State can directly affect the active quests. While performing quests, the Player receives help from the Player's Assistant (described in more detail in [35]), which monitors the player progression through the generated quest plans, providing him/her with *tips* about his/her next objectives and goals. The Quest Library contains a database of quests and sub-quests specified as planning problems, which are dynamically solved by the Quest Planners in real-time.

3.3. The prototype game

The game used to test and validate our method is a 2D RPG (Fig. 3) that uses the proposed architecture to dynamically generate and control the entire narrative of the game. The narrative pertains to a *zombie survival* genre and tells the story of a family that lives in a world dominated by a zombie plague. The player controls the brave husband John through several nondeterministic quests to protect his family and save his own life. The game is composed of 26 quests (8 deterministic and 18 nondeterministic) with different hierarchical levels and complexities. In the baseline story, John's wife is attacked by a zombie and is saved by John, who finds an antidote. Then, in order to defend his family, John tries to improve the protection of his house, but his daughter ends up being attacked by another zombie. After failing in protecting his house, John and his family escape to a remote island, where they have to build a house and find supplies to survive. Unfortunately, some zombies also find their way to the island and attack John and his family again. John survives the attack, but the future of his family is still uncertain. Several different stories with happy, sad, and even dark outcomes can emerge from this basic storyline depending on the player's personality, behavior and decisions while performing nondeterministic quests. John's wife and his daughter may survive or not, after being infected by a zombie, depending on whether the player succeeds in getting an antidote. After escaping to the island, the player may fail in the quest for supplies, and one or more members of his family may starve to death. John can even be unable to escape to the island if he fails in a quest to get fuel for his boat. This unfortunate event will force him to escape to a remote mountain, where a different story takes place.

The gameplay of the prototype game is driven by the story quests, wherein the player has to collect items, interact with non-player characters and kill enemies (zombies). In order to fight against the zombies, the player has a gun with a limited amount of ammunition, which is reloaded when the player collects ammunition kits. When the player is attacked by zombies, he/she loses an amount of life (i.e. of the life energy initially attributed to the player), which is only restored when he/she collects medic kits. In addition to the enemies, the player finds through the game two types of non-player characters: (1) normal non-player characters, which are characters that talk and interact with the player; and (2) non-player characters in dangerous situations, which are characters that can be saved by the player.

4. Player behavior model

Human behavior is the result of complex reflective-impulsive processes [36], which are influenced by a series of factors (e.g.: personality traits, beliefs, and cultural aspects). In addition, behavior is also dynamic, meaning that it is susceptible to vary with time, place, situation, and context [7]. A computational model built to predict player behavior must be able to handle, not only its complex nature, but also its natural

Fig. 3. Scene of the prototype game. The player's avatar (John), surrounded by zombies, is about to lose the antidote that can save infected members of his family.

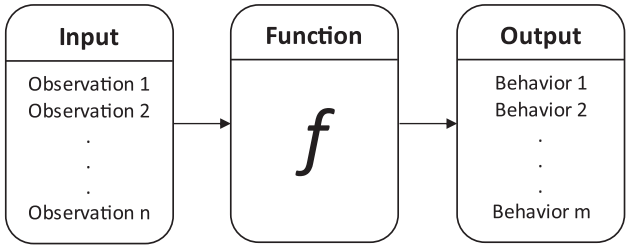

Fig. 4. Components of a general player behavior model.

dynamism.

As illustrated in Fig. 4, a general player behavior model is composed of three main components: input, output, and a function. The model's input comprises a set of observations extracted from the gameplay data, which should convey enough information about the player's behavior. The model's output represents the set of behaviors that can be predicted by the model based on the input observations. The model's function is the core of the model – it maps the input observations into the output behaviors. The next sub-sections describe how these general components are implemented in the proposed behavior model.

4.1. Observations (input)

Since behavior is dynamic and time-dependent, the model must be able to recognize all behavioral changes that occur over time. Consequently, the system must be constantly capturing gameplay data and using it as input to the model. This dynamic process is performed using *time windows*, which are constant time intervals where the gameplay data is collected and then used to predict the current player behavior. The length of the windows is a crucial variable to determine the precision of the model. Too short time windows may provide only a limited amount of information about the player behavior, but too long windows may fail in capturing transitions between behaviors and produce blurred data. In order to determine the best length for the time windows, we conducted several tests with window sizes of varying length. The results of these experiments are presented in Section 7.

The data used as input to a player model is composed of a set of statistical features extracted from the gameplay during a time window. Although these features are dependent of the game mechanics, we selected a collection of general gameplay features that can be found not only in our prototype game but also in other game genres, such as shooters, action-adventures, and role-playing games. In addition, our model of the features is fully *adaptive*, meaning that the features are computed according to the context from which they were extracted. For example, if a time window comprises the player's actions in a location with 8 enemies and another window includes his actions in a location with 2 enemies, all features related with the number of enemies must take this information into account. If the player kills all enemies in both time windows, the feature that represents the number of enemies killed must be the same for both cases.

The gameplay features used as input to our model are listed in Table 1, where T is the length of the time window in seconds, and the following indicators refer to what is seen by the player during the time window: E – total of enemies, A – total of ammunition kits, M – total of medic kits, N – total non-player characters, and D – total of non-player characters in danger.

4.2. Behaviors (output)

The next step to create a player behavior model is to define its output, consisting of the possible behaviors that the model will be able to predict in the future. The majority of previous works on player modeling adopt very simplified behavior models based on limited sets of player archetypes [17], which can be very restricted considering that players can exhibit interchangeable and unique blended behaviors. In addition, they often fail in providing more detailed information about the behaviors, such as their intensity.

To define a more general and robust output for our behavioral model, we adopted a widely accepted theory about human personality: the Five Factor Model (also known as "Big Five") [9]. Big Five is a dimensional representation of human personality structure, which claims that, by using five personality traits, it can suitably account for personality diversity. The Big Five factors are:

(1) **Openness:** those who are high on this factor are imaginative, curious and open to new ideas. In contrast, those who score low on this factor are indifferent and uninterested;
(2) **Conscientiousness:** the ones that display high degree of this factor are meticulous, efficient and systematic. Who scores low is careless, chaotic and disorderly;
(3) **Extraversion:** high scorers are characterized by enjoying social activities. On the opposite side, low scorers are reserved and shy.
(4) **Agreeableness:** a high score on this factor characterizes helpful, cooperative and friendly people. In contrast, low score characterizes selfish and hostile people.
(5) **Neuroticism:** those who score high on this factor are emotionally unstable, anxious and aggressive. In contrast, those who score low are well-adjusted and calm

The five dimensions of the human personality structure are supported by several questionnaires, inventories, and adjective rating scales designed to measure each dimension (e.g.: [37–39]). Personality classification is then achieved by assigning five numerical scores (one per dimension) that account for how well each factor describes the person. The attribution of the scores is typically performed with questionnaires that consider observable behavior and characteristics of the individual.

Although the Big Five factors are ordinarily used to describe individual personality, they can also be correlated with specific behaviors. In fact, past studies have pointed that several human behaviors can be adequately explained in terms of the Five Factor Model [40,41]. One example of behavioral taxonomy based on the Big Five is presented by Back et al. [40], who assigned a multitude of concrete actual behaviors to each of the five dimensions of the Big Five on the basis of a systematic investigation of theoretical and empirical approaches to personality and social behavior.

The output of our model is represented by the Big Five factors, which are disposed on five behavioral axes (Fig. 5), each within the interval of $[-1, 1]$. We used the behavioral aspects proposed by Back et al. [40] to define the general behavioral aspects of each factor (Table 2), which we divided into positive $(+)$ and negative $(-)$ behaviors in accordance with the factor's score. The sign $(-$ or $+)$ does not mean destructive or constructive behaviors, but simply indicates the two opposite sides of the Big Five dimensions (i.e. low and high scores). Then we associated each behavioral aspect with a set of general in-game player behaviors, which describes concrete player behaviors within a game environment. For example, if a player is curious and interested, he/she should demonstrate this behavior by exploring the environment. Also, we expect that a meticulous, systematic and efficient player rarely gets attacked by enemies, hardly misses a shot, and only collects and uses items when they are needed. These in-game player behaviors depend on the particular type of game within a specific game genre. Furthermore, we must acknowledge the theoretical and practical limitations of identifying personality through in-game behavior, which require further research. However, the idea of building relationships like the ones presented in Table 2 is a powerful basis for player behavior modeling.

For the purpose of presenting the algorithm in Section 6.1, we define 5 functions that represent the behavior of a player p at time t:

Table 1
Gameplay features.

ID	Description
F_1	Percentage of time that the player is standing still (in relation with T)
F_2	Percentage of time that the player is walking (in relation with T)
F_3	Percentage of time that the player is colliding (in relation with T)
F_4	Total of new areas explored by the player
F_5	Total of shots fired by the player during the time window
F_6	Percentage of shots that hit targets (in relation with F_5)
F_7	Percentage of shots that miss targets (in relation with F_5)
F_8	Percentage of enemies killed by the player (in relation with E)
F_9	Average time interval between shots fired by the player
F_{10}	Standard deviation of the time intervals between shots fired by the player
F_{11}	Average distance in which enemies were killed by the player
F_{12}	Standard deviation of the distances in which enemies were killed by the player
F_{13}	Average distance in which enemies were hit by shots fired by the player
F_{14}	Standard deviation of the distances in which enemies were hit by shots fired by the player
F_{15}	Average time spent by the player to kill enemies after seeing them
F_{16}	Standard deviation of the times spent by the player to kill enemies after seeing them
F_{17}	Percentage of medic kits collected by the player (in relation with M)
F_{18}	Percentage of life the player recovered without necessity (i.e. by using medic kits when the player's life was almost full) (percentage calculated in relation with F_{17})
F_{19}	Average time spent by the player to collect medic kits after seeing them
F_{20}	Standard deviation of the times spent by the player to collect medic kits after seeing them
F_{21}	Percentage of ammunition kits collected by the player (in relation with A)
F_{22}	Percentage of ammunition kits used by the player without necessity (i.e. by using ammunition kits when the gun clip was almost full) (percentage calculated in relation with F_{21})
F_{23}	Average time spent by the player to collect ammunition kits after seeing them
F_{24}	Standard deviation of the times spent by the player to collect ammunition kits after seeing them
F_{25}	Percentage of non-player characters with whom the player interacted and talked (in relation with N)
F_{26}	Average time spent by the player to talk with non-player characters after seeing them
F_{27}	Standard deviation of the times spent by the player to talk with non-player characters after seeing them
F_{28}	Percentage of non-player characters in danger saved by the player (in relation with D)
F_{29}	Average time spent by the player to save non-player characters in danger after seeing them
F_{30}	Standard deviation of the times spent by the player to save non-player characters in danger after seeing them
F_{31}	Total damage suffered by the player during the time window (i.e. total life loss)
F_{32}	Total of times the player changed his direction during the time window

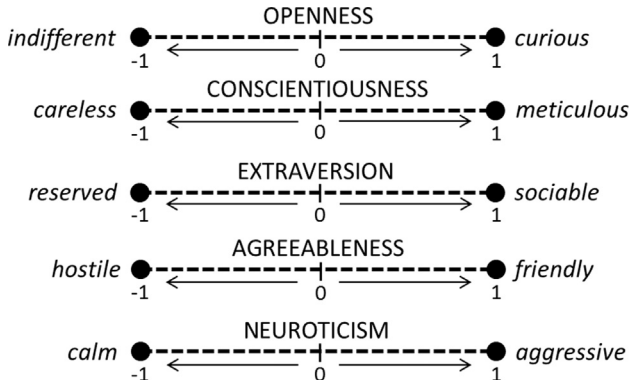

Fig. 5. Big Five factors represented as behavioral axes.

$$b_o = oppeness\_bhv(p, t) \quad (1a)$$

$$b_c = conscientiousness\_bhv(p, t) \quad (1b)$$

$$b_e = extraversion\_bhv(p, t) \quad (1c)$$

$$b_a = agreeableness\_bhv(p, t) \quad (1d)$$

$$b_n = neuroticism\_bhv(p, t) \quad (1e)$$

We use logic predicates to represent the behavior functions above, such as $oppeness\_bhv(p, t, b_o)$ to represent $b_o = oppeness\_bhv(p, t)$. Furthermore, in our implementation, the time t is omitted because time is a discrete variable controlled by the state updating mechanism.

4.3. Behavior predictive function

Once we have defined the input and output of our model, we need to establish a function capable of learning and predicting player

Table 2
General behavioral aspects of the Big Five factors.

Big Five Factors	Behavioral aspects	In-game player behavior
Openness	+ curious, interested, inquisitive − indifferent, incurious, uninterested	• Explores the environment • Collects all the available items • Explores only indispensable parts of the environment • Collects only indispensable items
Conscientiousness	+ meticulous, efficient, systematic − careless, chaotic, disorderly	• Rarely gets attacked by enemies • Rarely misses a shot • Collects and uses items only when they are needed • Frequently gets attacked by enemies • Frequently misses shots • Collects and uses items when they are not needed
Extraversion	+ sociable, talkative, active − reserved, shy, passive	• Frequently interacts with non-player characters • Interacts with non-player characters as soon as possible • Rarely interacts with non-player characters • Postpones interactions with non-player characters
Agreeableness	+ friendly, altruistic, helpful − hostile, selfish, obstinate	• Always tries to save non-player characters that are in danger • Rarely tries to save non-player characters that are in danger
Neuroticism	+ aggressive, nervous, unstable − calm, relaxed, balanced	• Tries to kill all enemies • Performs disordered movements • Kills only threatening enemies • Performs only necessary movements

behaviors. Considering that the output of our model comprises five numerical values representing the Big Five factors, the task to build this function can be seen as a multi-output regression problem [42]. Existing methods to handle this type of problem can be categorized as: (1) problem transformation methods, where the multi-output problem is converted into independent single-output problems, which are solved using a single-output regression algorithm; and (2) algorithm adaptation methods, which adapt single-output methods to directly handle the multi-output data. Among these methods are those using Artificial Neural Networks [43], which work as a typical multi-output regression algorithm to handle problems where the outputs are independent of each other.

In the proposed system, we implemented the model's function using an Artificial Neural Network trained to predict the values for the Big Five factors based on the statistical features extracted from the gameplay (Table 1). More specifically, we employed a single hidden layer Neural Network trained by an incremental back-propagation learning algorithm using a sigmoidal activation function. In our experiments, we used 64 neurons in the hidden layer. The algorithm was implemented using the FANN library.[3]

Since our method employs a supervised machine learning technique to create the player model, samples of gameplay sessions need to be captured and annotated by an expert with labels describing the current player behavior. Considering that our model characterizes the player behavior with five numerical scores, each representing one of the Big Five factors, this annotation process must cover all characteristics of the observable behavior and measure each score systematically. To standardize this process and assist the human expert during the annotation process, we created a simple training questionnaire based on the general in-game player behaviors that contribute to the scores of each Big Five factor in accordance with the rating scales of the Revised NEO Personality Inventory (NEO PI-R) [38]. Our questionnaire is composed of 10 statements regarding the observable player behavior. Each statement is followed by a ten-point Likert scale on which the expert has to rate how much of the behavior indicated by the statement he/she could observe on the analyzed gameplay segment. The scale ranges from "not even a little" (−5) through "neutral" (0) to "a lot" (5). Each statement contributes to the measurement of at least one of the Big Five factors used to characterize the player behavior. The full questionnaire is available in a separate online document.[4]

After capturing the data, each sample (the set of features for a particular time interval) was associated with its respective video segment (i.e. the video segment extracted from the game screen video at the exact time interval during which the features were captured). Then, three voluntary human experts analyzed all video segments and used the training questionnaire to measure and annotate the scores of the Big Five factors for the samples of each time window. We selected these experts from a group of computer science students capable of analyzing games as experienced players. We have avoided professionals in player behavior, not because they are hard to find, but mainly because we want tasks and questionnaires that can be easily completed by simple observation of superficial actions of the players.

After creating and selecting the best dataset, the Neural Network can be trained offline and then used to predict the player behavior in real-time. An evaluation of the precision and performance of the Neural Network and further details of our experiments can be found in Section 7.1.

5. Player personality model

While behavior is susceptible to variation in consequence of changes in time, place and context [7], personality traits are relatively enduring, as remarked by Costa and McCrae [44], who define personality as a combination of characteristics that form a distinctive character, an individual style of thinking, feeling and acting. According to Back and Egloff [45], personality arises from interactions between the situation in which the individual is placed and the processes that take place inside the individual's mind.

Personality plays an important role in influencing player's preferences for game genres [46], characters [47], and narratives [8]. Having some information about the current player's personality and preferences can help the game to adapt its content and create personalized and satisfying playing experiences. While traditional forms of storytelling have no access to this information, the interactive nature of games gives players the opportunity to convey their preferences (consciously or not) through the actions that they perform as part of the game interaction.

The personality of an individual can be determined through a variety of tests and measurement scales. Among the most widely accepted are those that follow the Big Five proposal [9]. However, differently from the player *behavior model* (Section 4), where the Big Five factors were used to describe in-game player behaviors, a *personality model* must describe the personality of the players in the real world, which may not be in agreement with their behavior in the game world. Another difference between the two models is the time when they need to be computed and updated: while the behavior model can be gradually constructed and updated during the gameplay, the personality model must be established before the beginning of the main storyline, which allows the game to correctly adapt the whole narrative according to the player's personality. In our system, the personality model is used in two moments (see Section 6.1): setting up goal hierarchies and defining planning operators.

5.1. Strategies for personality assessment

The Big Five dimensions are usually assessed through long questionnaires (60 or 44 items). In interactive storytelling systems and games, we need special care. First we should avoid forcing players to answer non-game related questions, because this may have negative effects on the player experience. Instead, a better solution is to adapt and integrate questionnaire statements into the game through story-related dialog choices, so as to reduce the players' uncomfortable awareness that their personality is being measured. Secondly, we should finish the assessment very quickly. Therefore it would be helpful to adopt simplified questionnaires, such as the BFI-10 [48], which is one of the shortest questionnaires that measures the scores of the Big Five factors with only 10 questions.

Another well-known short measurement scale is the *Ten-Item Personality Inventory* (TIPI) proposed by Gosling et al. [49], which also uses two items associated with each personality dimension. However, in this work, we favored BFI-10 over TIPI because of the following reasons: (1) BFI-10 uses a five-point Likert scale rather than the seven-step scale of TIPI – which makes BFI-10 simpler and faster (although both take about a minute to complete); (2) BFI-10 uses statements representing two extremes of the same dimension clearly, which are more aligned with actions and attitudes than the more generic opposite adjectives of TIPI[5] – this fact makes BFI-10 more adequate for the lively situations of the game scenes used in the present work; (3) the BFI-10's authors [48] have shown that BFI-10 is psychometrically superior to TIPI; (4) BFI-10 was successfully tested in more than one idiom, besides the original version in English and German [50,51] – this fact suggests

[3] Fast Artificial Neural Network Library - http://leenissen.dk/fann/.

[4] http://www.icad.puc-rio.br/~logtell/interactive-quests/training_questionnaire.pdf.

[5] As an example, for the agreeableness dimension, BFI-10 evaluates how much one "is generally trusting" and "tends to find fault with others", while TIPI rates the extent to which the following adjectives apply to you: "critical, quarrelsome" and "sympathetic, warm".

that BFI-10 might be a better alternative to TIPI and, more importantly, be more adequate for multi-language games.

In BFI-10, the subject answers the following 10 questions "I see myself as someone who …": (1) is reserved; (2) is generally trusting; (3) tends to be lazy; (4) is relaxed, handles stress well; (5) has few artistic interests; (6) is outgoing, sociable; (7) tends to find fault with others; (8) does a thorough job; (9) gets nervous easily; (10) has an active imagination. The answers (L values) are given in a five-point Likert scale: 1 (disagree strongly), 2 (disagree a little), 3 (neither agree nor disagree), 4 (agree a little), and 5 (agree strongly).

For each Big Five dimension, BFI-10 calculates the average score of two poles, which correspond to a true-scored item and a false-scored item respectively. The false-scored item must be reverse scored before calculations are made, i.e. the values 1, 2, 3, 4, and 5 become 5, 4, 3, 2, and 1 respectively, or mathematically:

$$reversed\ score = 6 - score \tag{2}$$

For instance, for the Neuroticism dimension, BFI-10 evaluates how much one "gets nervous easily" (say, $L = 4$ for question 9) and "is relaxed, handles stress well" (say, $L = 5$ for question 4), that is, Neuroticism is $(4 + 1)/2 = 2.5$ (where 1 is the reversed score of 5). The scored items for each dimension are defined as follows (where R indicates a reversed-scored item): Extraversion (6 and 1R), Agreeableness (2 and 7R), Conscientiousness (8 and 3R), Neuroticism (9 and 4R), and Openness (10 and 5R).

5.2. A personality inventory for games: BFGI-10

In order to assess players' personalities in our prototype game, we created a new Big Five inventory based on the BFI-10. The proposed inventory, which we called Big Five Game Inventory (BFGI-10), comprises more than just questions and measurement scales, it includes 10 story-related scenes followed by decision-making points (one for each BFI-10 question), where players make decisions that are equivalent to answering BFI-10 questions. Each scene creates a situation that stimulates players to react in a way that makes evident their answer to the BFI-10 question that defined the scene. All scenes are presented to players at beginning of the game as a single interactive cutscene. In our zombie survival game, the cutscene tells how the zombie plague started from the viewpoint of the main character (John, the housefather). At the end of each scene, the player must inform how he/she would react to the presented situation by choosing between five options, which are equivalent to the Likert scale of BFI-10, but with descriptions that are related with the scene. For example, for the BFI-10 question 9 ("I see myself as someone who gets nervous easily"), we created a scene where a clumsy colleague of John accidently spills coffee on John's shirt. After a short dialog, the game asks what the player would do if he/she were John (Fig. 6). In this case, the five options that are equivalent to answer the BFI-10 question 9 are: do nothing ($L = 1$), forgive Ted ($L = 2$), ask for Ted's apologies and then forgive him ($L = 3$), get nervous and rebuke Ted ($L = 4$), and get very nervous and strongly rebuke Ted ($L = 5$). The full description of the BFGI-10 scenes and questions used in our prototype game is available in a separate online document.[6]

In order to compute the final scores of the Big Five dimensions in BFGI-10, we normalize the score bf_i of the i-th dimension in the interval [0, 1] instead of [1,5], i.e.:

$$bf_i = \frac{\overline{bf_i} - 1}{4} \quad i = 1, 5 \tag{3}$$

where

$$\overline{bf_1} = \overline{bf}_{extraversion} = \frac{L_6 + L_1^R}{2} \tag{4a}$$

$$\overline{bf_2} = \overline{bf}_{agreeableness} = \frac{L_2 + L_7^R}{2} \tag{4b}$$

$$\overline{bf_3} = \overline{bf}_{conscientiousness} = \frac{L_8 + L_3^R}{2} \tag{4c}$$

$$\overline{bf_4} = \overline{bf}_{neuroticism} = \frac{L_9 + L_4^R}{2} \tag{4d}$$

$$\overline{bf_5} = \overline{bf}_{openness} = \frac{L_{10} + L_5^R}{2} \tag{4e}$$

and L_j and L_k^R are the Likert scale values of the true-scored item and the reversed-scored item of each dimension respectively. For the purpose of presenting the algorithm in Section 6.1, we define 5 *constant functions* that are simply referred by their names, which represent the personality traits of a player p:

$$openness = openness(p) = \overline{bf}_{openness} \tag{5a}$$

$$conscientiousness = conscientiousness(p) = \overline{bf}_{conscientiousness} \tag{5b}$$

$$extraversion = extraversion(p) = \overline{bf}_{extraversion} \tag{5c}$$

$$agreeableness = agreeableness(p) = \overline{bf}_{agreeableness} \tag{5d}$$

$$neuroticism = neuroticism(p) = \overline{bf}_{openness} \tag{5e}$$

Moreover, in our implementation, we use predicates to represent the constant functions above, such as $openness(p, \overline{bf}_{openness})$ to represent $oppeness(p) = \overline{bf}_{openness}$.

The ability of BFGI-10 to determine player's personalities depends on how well the proposed story-related scenes and questions simulate and represent the BFI-10 scales. An evaluation on this matter is presented in Section 7. Although BFGI-10 was specifically designed for our game narrative, its scenes and questions can be easily adapted and used as basis for the development of interactive cutscenes to assess the personality of players in other game genres.

6. Quest manager

As presented in Section 3.1, the Quest Manager is composed by a *Quest Planner*, which generates hierarchical quests and logical plans of actions, and a *Quest Monitor*, which monitors changes introduced by the player.

6.1. Quest planner

6.1.1. The proposed algorithm

We define a *quest* as a planning problem, expressed by the tuple[7]:

$$Q = (P, S_0, G, H, O)$$

where P is a set of *atomic formulas* (or *atoms*, for short), O is a set of planning operators that may require other quests, S_0 is the initial state, G is a set of alternative authorial goals, and H represents the authorial goal hierarchies associated with specific personality traits, such that:

- An *atom* is an expression of the form $pred(r_1, \cdots, r_k)$, such that *pred* is a predicate symbol and $r_1, \cdots, r_k$ are variable terms (e.g. `CH1` and `IT`) or ground terms (e.g. `player` and `antidote`);
- A *literal* is an atom p or the negation of an atom, $\neg p$, letting negation signify the deletion of the proposition from the current world state S (i.e. we use the *closed-world assumption*: a proposition that is not explicitly specified in a state does not hold in that state);
- $S_0 \subseteq P$ is a set of ground literals;
- $G = \{G_1, G_1, \cdots, G_n\}$, where each goal $G_i \subseteq P$ is a set of ground literals;

[6] http://www.icad.puc-rio.br/~logtell/interactive-quests/bfgi-10.pdf.

[7] Notational conventions closely adapted from [10].

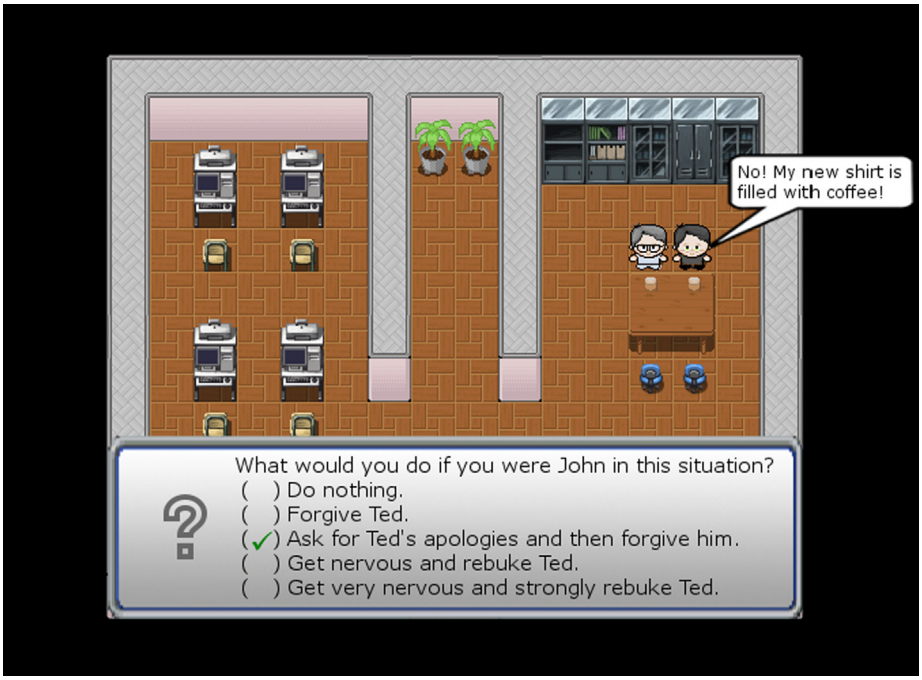

Fig. 6. Decision-making point in the introductory scene representing the BFI-10 statement 9 ("I see myself as someone who gets nervous easily").

- H is a set of pairs $H_i = \langle C_i, T_i \rangle$, to be read as "if C_i then T_i", where C_i is a Boolean expression using one or more of the personality trait functions (Eqs. (5), defined at the end of Section 5.1) to represent a *personality condition*, and T_i is a totally ordered subset of G under the relation $t_i \prec t_j$, meaning that the attempt to achieve the goal t_i must occur before the attempt of reaching the alternative goal t_j. Therefore, the personality conditions C_i use the player's personality traits found by the system during the preliminary session with story-related scenes (i.e. during the application of our BFGI-10 method described in Section 5.1). In our system, this is the first place where the narrative is adapted according to the player's personality. For the sake of simplicity, we only use AND-relations in the Boolean expressions of personality conditions C_i. For example, let $G_1 = \{dead(anne)\}$, $G_2 = \{healthy(anne), protected(house)\}$ and $G_3 = \{escaped(john)\}$ be the alternative goals G, and suppose that the hierarchy H_1 is $C_1 = \{agreeableness \geq 0.4, conscientiousness > 0.8\}$ and $T_1 = \{G_2, G_1, G_3\}$. This means that "if agreeableness $\geq$ 0.4 and conscientiousness $>$ 0.8 then $G_2 \prec G_1 \prec G_3$", that is: if the conditions C_1 are satisfied, then the planner first will try to achieve G_2, and if failure occurs then the planner will try G_1 and, finally, G_3 if G_1 fails.

Using a common notation in planning theory6, the operator $o \in O$ is denoted by the 4-tuple:

$$o = (name(o), precond(o), effect(o), subq(o))$$

where:

- $name(o)$ is the name of the operator, which is an atom $op(x_1, x_2, \cdots, x_k)$, where op is a unique symbol called an operator symbol, and x_i is a variable symbol that occurs anywhere in o;
- $precond(o)$ is a set of literals that define the preconditions of o (i.e. the literals that must be true in order to use the operator o). Player behavior functions (Eqs. (1)) and personality trait functions of the player (Eq. (5)) are important terms of these preconditions. The first set of functions represent dynamic values predicted by the model at time t. The latter set of functions represent constant values obtained during the preliminary session with story-related scenes. Instead of using current values of player behaviors, we use average values as explained in Section 6.1.2. This is the first place in the system where we use player behaviors, but this is where we use personality traits for the second time;
- $effect(o)$ is a set of literals that define the effects of o (i.e. the literals the operator o will make true);
- $subq(o)$ is the quest required by o, which we call *sub-quest* of o.

When $subq(o)$ is not empty, o is referred as a compound operator otherwise it is a primitive operator. An action a is any ground instance of a planning operator o. The following definitions apply to actions:

- An action a is applicable to the current world state S if the preconditions of a hold in S.
- An action a is relevant for a goal G_i (i.e. a can produce a state that satisfies G_i) if the effects of a hold in G_i and are consistent with one of the goals of $subq(a)$. This condition is satisfied when the world state generated by the effects of a is the initial condition of $subq(a)$ and the sub-quest succeeds.

An instance of a compound operator is a total-order plan (i.e. a totally ordered sequence of actions), resulting from the concatenation of the solutions of all sub-quests down the hierarchy. Each sub-quest is handled as a classical planning problem. We consider the cost of doing an action a in a state s as unitary, i.e. $cost(a, s) = 1, s \in S$.

The following examples in a zombie survival game illustrate the hierarchical quests:

```
quest: save-family
s0: character(john), character(anne), place(home),
   place(forest), at(john,forest), at(anne,forest),
   healthy(john), infected(anne), safe(home), path
   (forest,home), path(home,forest)
G1: healthy(anne), protected(house)
G2: dead(anne), escaped(john)
H1: if {agreeableness ≥ 0.4} then G1 ≺ G2
H2: if {agreeableness < 0.4} then G2 ≺ G1

Operator: take(CH1, CH2, PL1, PL2)
```

```
precond: healthy(CH1), infected(CH2), at(CH1,PL1),
   at(CH2, PL1), path(PL1, PL2), CH1 ≠ CH2
effects: ¬at(CH1, PL1), ¬at(CH2, PL1), at(CH1,PL2),
   at(CH2, PL2)
subq: ∅

Operator: save(CH1, CH2, PL)
precond: healthy(CH1), infected(CH2), at(CH1, PL), at
   (CH2, PL), safe(PL), CH1 ≠ CH2
effects: healthy(CH2), ¬infected(CH2)
subq: save-wife

Operator: protect(CH, PL)
precond: healthy(CH), at(CH, PL)
effects: protected(PL)
subq: protect-house

Operator: abandon(CH1, CH2, PL)
precond: healthy(CH1), infected(CH2), at(CH1, PL), at
   (CH2, PL), safe(PL), CH1 ≠ CH2
effects: dead(CH2)
subq: abandon-wife

quest: save-wife
s0: character(john), character(anne), character
   (oldman), place(home), place(village), place
   (hospital), place(market), item(antidote1), item
   (antidote2), at(john,home), at(anne,home), at
   (oldman,market), healthy(john), infected(anne),
   at(antidote1,hospital), has(oldman,antidote2),
   safe(home), path(home,village), path
   (village,home), path(village,hospital), path
   (hospital,village), path(market,village), path
   (village,market)
G1: healthy(anne)
G2: dead(anne)
H1: if {agreeableness ≥ 0.4} then G1 ≺ G2
H2: if {agreeableness < 0.4} then G2 ≺ G1
```

In the above examples we can notice that `john`, `anne`, and `oldman` are characters; `house`, `village`, `market`, `forest`, and `hospital` are places; `john` and `anne` are both at `home`; `john` is healthy, but `anne` is infected; `antidote` is an item that is at the `hospital`; and there is a path connecting `home` with the `hospital` (amongst other paths connecting places). Also we can see that if the compound operator `save` is instantiated as `save(john, anne, home)`, the sub-quest `save-wife` will be triggered, because one of its goals (namely `healthy(anne)`) is one of the effects of the action `save(john, anne, home)`.

Compound operators represent nondeterministic events that may have different effects on the story plot depending on the player's interferences and decisions while the quest monitor is performing the sub-quests. Although the compound operators may have nondeterministic effects on the quest plan, they are specified with a default list of deterministic effects according to the primary authorial goal of its respective sub-quests. The nondeterministic nature of the compound operators is handled by the Quest Monitor in real-time during the execution of the quest plan.

The game world is logically represented by a state, which consists of a set of ground propositions $S \subseteq P$ defining characters, objects, locations, and their current situation in the game world. If a sub-quest is called, the current state of the world will be used as the initial state of this new sub-quest. Therefore, when the player causes changes in the world, the planner recalculates the quest plan using the modified world state as the initial state S_0 of a new classical planning problem. The proposed algorithm can use any classical planner for this step of a simple quest. In our implementation, we used the HSP2 planner provided by Bonet and Geffner [52], which is fully compatible with our STRIPS-like formalism.

Any sub-quest is described as an independent planning problem in the Quest Library. The Quest Planner adopts a hierarchy of authorial goals, in the sense that, if the intended goal cannot be achieved, the planner tries its immediate successor in the authorial goal hierarchy. The planner can fail to achieve a desired goal either if there is no valid sequence of actions that leads from the initial state to the goal state, or if the prescribed time limit for searching for a solution is exceeded. In both cases, as already remarked, the planner tries to achieve the next successor goal from the authorial goal hierarchy.

For example, the quest `save-family` has two different outcomes (G_1 and G_2) with two possible authorial goal hierarchies (H_1 and H_2). In this example, the alternative goal hierarchy is selected based only on the player's score on the agreeableness personality factor. If the player's score is higher than 0.4 (meaning that player's personality indicates someone who is helpful, cooperative and compassionate), then the hierarchy of goals H_1 is selected as the one that best match the player's personality. In H_1, G_1 is the primary goal of the quest, which establishes that `anne` must be healthy and the `house` must be protected (i.e. the player must save John's wife and protect his house). If the player modifies the game world in such a way that G_1 becomes unreachable, the planner will try to find a plan to achieve G_2, which requires `john` to save himself escaping from the zombies. At a lower hierarchic level concerning the same example, if the first goal of `save-wife` (which is a sub quest of `save-family`) fails, then the second goal may be pursued, which would compel the husband to kill his wife to save her from the doom of being a walking mindless monster forever. As a consequence, the first requirement to achieve G_2 (`anne` must be dead) will be automatically accomplished.

Otherwise, if the player's score on the agreeableness factor is lower than 0.4 (which indicates someone who is selfish and hostile, and lacks empathy), then the hierarchy of goals H_2 will be selected. In H_2, G_2, which is the primary goal of the quest, establishes that `anne` must be dead and `john` must escape (i.e. to complete the quest, the player has to abandon John's sick wife to a certain death and escape from the zombies). Again, if G_2 becomes unreachable, the planner will try to find a plan to achieve the next goal in the hierarchy (in this case G_1). As a general authorial rule, it should be ensured that, in any situation that may be predicted, at least one goal in every H_i set should be achievable to avoid aborting the story prematurely.

The definition of the personality conditions for the possible orders of alternative goals is based on a personality test that is conducted with prospective players during the quest design process. In this test, a group of potential players answer a traditional BFI-10 questionnaire to determine their personality. Then, they rate textual descriptions of the outcomes of each quest according to their preferences. The rates are statistically correlated with the personality traits to establish the orders of alternative goals that best match the preferences of the major groups of players' personalities, which generates a set of personality conditions that are used in the alternative goal hierarchies. In our experiments, the personality test was conducted with a group of 27 potential players and, even with this small number of subjects, we were able to identify some useful preference patterns based on the personality scores.

Once a quest has started, the planning algorithm proceeds by searching in the space of world states for a sequence of actions that leads the player from the current state of the world to one of the quest's goals. However, differently from a traditional HTN planning algorithm, and to improve the performance of the planner, our algorithm does not decompose the compound operators during the generation of the initial plan for the quest, which would significantly affect the performance of the planner. The planner interprets compound operators as primitive and uses their predefined deterministic effects to generate a plan without instantiating the events of sub-quests. When the player reaches a compound operator, the plans for sub-quests are generated by new instances of quest planners. In this way, our algorithm deals with nondeterminism efficiently and gracefully.

6.1.2. Player behaviors as preconditions

The player behavior functions (Eqs. (1)) could be used as part of the preconditions for both primitive and compound operators. The values of these functions refer to the last player behaviors observed (predicted by the model) at time T. However, average scores accumulated over time for the current player represent a more adequate approach to write general rules. Therefore, instead of using current values of player behaviors (Eqs. (1)), we use their average values calculated as *means* over the interval $[0, T]$:

$$\overline{b}_o = \frac{1}{T}\int_0^T oppeness\_bhv(p, t)dt \tag{6a}$$

$$\overline{b}_c = \frac{1}{T}\int_0^T conscientiousness\_bhv(p, t)dt \tag{6b}$$

$$\overline{b}_e = \frac{1}{T}\int_0^T extraversion\_bhv(p, t)dt \tag{6c}$$

$$\overline{b}_a = \frac{1}{T}\int_0^T agreeableness\_bhv(p, t)dt \tag{6d}$$

$$\overline{b}_n = \frac{1}{T}\int_0^T neuroticism\_bhv(p, t)dt \tag{6e}$$

However, considering that we are using time as a discrete variable within a logic formalism, we calculate arithmetic means and represent these functions by logic predicates, where time t is implicit:

$avg\_openness\_bhv(p, \overline{b}_o)$

$avg\_conscientiousness\_bhv(p, \overline{b}_c)$

$avg\_extraversion\_bhv(p, \overline{b}_e)$

$avg\_agreeableness\_bhv(p, \overline{b}_a)$

$avg\_neuroticism\_bhv(p, \overline{b}_n)$

We could use *medians* instead of means, because the median is less affected by outliers and skewed data. However, we left this approach for future experimental trials.

The following examples illustrate the usage of different player behaviors as part of the precondition for operators:

```
Operator: give(CH1, CH2, IT, PL)
precond: at(CH1,PL), at(CH2,PL), healthy(CH2), has
   (CH1,IT), CH1 ≠ CH2, avg-agreeableness-bhv
   (CH2,X), X ≥ 0.5
effects: has(CH2,IT), ¬has(CH1,IT)
subq: Ø

Operator: not-give(CH1, CH2, IT, PL)
precond: at(CH1,PL), at(CH2,PL), healthy(CH2), has
   (CH1,IT), CH1 ≠ CH2, avg-agreeableness-bhv
   (CH2,X), X < 0.5, avg-neuroticism-bhv(CH2,Y), Y <
   0.5
effects: ¬has(CH2,IT)
subq: Ø

Operator: kick-out(CH1, CH2, IT, PL)
precond: at(CH1,PL), at(CH2,PL), healthy(CH2), has
   (CH1,IT), CH1 ≠ CH2, avg-agreeableness-bhv
   (CH2,X),X < -0.5, avg-neuroticism-bhv(CH2,Y), Y ≥
   0.5
effects: ¬has(CH2,IT)
subq: Ø
```

The examples show three different operators that can occur after an `ask` event, where the player (`CH2`) asks another character (`CH1`) for an item (`IT`) in a specific place (`PL`). The operator `give` can only occur if the player has been behaving in a friendly manner towards the others in the game (with an average score of the agreeableness factor higher than 0.5). Otherwise, if the average score of the agreeableness factor be lower than 0.5 and the average score of the neuroticism factor also be lower than 0.5 (meaning that the player is not being very friendly, but is not behaving aggressively either), the operator `not-give` can occur. However, if the player has been hostile (agreeableness factor lower than -0.5) and aggressive (neuroticism factor higher than 0.5) with others, then `kick-out` is the only applicable operator.

6.1.3. Player's personality traits as preconditions

Just like player behavior, the player's personality traits (Eqs. (5)) can also be included in the precondition for both primitive and compound operators. The following examples illustrate the usage of the player's personality in the precondition for operators:

```
Operator: tell-backstory-detailed(CH1, CH2, PL)
precond: at(CH1,PL), at(CH2,PL), healthy(CH1),
   healthy(CH2), CH1 ≠ CH2, has-backstory(CH1),
   openness(CH2,X), X > 0.5
effects: know-backstory(CH2,CH1)
subq: Ø

Operator: tell-backstory-short(CH1, CH2, PL)
precond: at(CH1,PL), at(CH2,PL), healthy(CH1),
   healthy(CH2), CH1 ≠ CH2, has-backstory(CH1),
   openness(CH2,X), X ≤ 0.5
effects: know-backstory(CH2,CH1)
subq: Ø
```

The above examples show two versions of the operator `tell-backstory`, where a character `CH1` tells his backstory to another character `CH2` in a place `PL`. Both versions have the same logical effect (`CH2` gets to know the backstory of `CH1`), but different preconditions and visual results. While the `tell-backstory-detailed` presents a long dialog scene about the backstory of `CH1`, the `tell-backstory-short` operator presents a shortest version of the dialog with only the information that is indispensable to understand the narrative. As for their preconditions, the `tell-backstory-detailed` operator occurs only when the player's personality score on the openness factor is higher than 0.5, which means that the player is curious, imaginative and probably would be interested in knowing more details about another character's backstory. In contrast, the `tell-backstory-short` operator occurs when the player's score on the openness factor is lower or equal to 0.5, which means that the player has a tendency to be indifferent and uninterested.

6.2. Quest Monitor

The Quest Monitor works together with the Quest Planner to generate and maintain the coherence of quests in the dynamic and non-deterministic environment of the game (Section 3.1). For each instance of a Quest Planner, there is a Quest Monitor in charge of monitoring the execution of its respective quest plan. Its job is to verify the occurrence of changes introduced by the player in the game world that violate preconditions of the quest events generated by the planner. In addition, the Quest Monitor is also responsible for instantiating new Quest Planners and Monitors to handle sub-quests described by the compound operators present in its respective quest plan.

The algorithm for monitoring the execution of quests continuously checks the current state of the world and the player model to verify the consistency of the quest plan. If it detects that the current world state is different from the expected state described in the quest plan, it requests a new plan from the Quest Planner using the current state of the world as the initial state for the planning problem. Similarly, if changes in the current or in the average player behavior are detected, a new plan is

requested. In this way, a new plan to achieve one of the quest goals will be generated. In this plan, new tasks may be added in order to make the player return to the previous storyline of the quests or a completely different sequence of events may be created to guide the quests towards a different outcome.

During the execution of the quest plan, the monitoring algorithm also verifies the occurrence of compound operators in the plan. If the next expected event of the quest is defined by a compound operator, a new Quest Planner and a new Quest Monitor are instantiated to handle the execution of the sub-quest independently. While the player is performing a sub-quest, the Quest Monitor of its parent quest waits until the player has finished the sub-quest to resume the monitoring process.

The process of monitoring the execution of quests, with the capacity of replanning the quest's events whenever necessary, allows the system to directly support nondeterministic sub-quests with multiple endings so as to influence the whole narrative of the game. In nondeterministic sub-quests, player's actions can induce the quest to an outcome that may affect the world state in a way that does not match the state produced by the predefined deterministic effects of its respective compound operator. Consequently, nondeterministic sub-quests can introduce inconsistencies in the plan of their parent quests depending on the way they end. Such inconsistencies will be automatically detected by the Quest Monitor, which will request a new plan to its respective Quest Planner, in order to correct inconsistencies and maintain the flow and coherence of the game. In this way, while performing a sub-quest, the choices made by the player are propagated through the hierarchy of quests, effectively modifying the game's narrative.

Fig. 7 illustrates how player actions and behaviors can modify the plot of quests, and how the combination of planning and monitoring can support nondeterministic quests and handle inconsistencies introduced by player's interventions in the plan of quests. In this example, the player is in a quest to save the life of his family, after his wife was attacked and infected by zombies.

Plan 1 describes the initial plan generated to solve the quest "*Save family*", which consists of taking the player's wife back home, saving her from the Zombie disease, and protecting his house. The sub-quest "*Save wife*" consists of going to the city hospital, getting the antidote, going back home, and using the antidote to save the wife's life. Suppose that, when the player is trying to go back home with the antidote, he is attacked by a zombie and breaks the antidote bottle. In this case, the fact `has(player, antidote)` will be removed from the current state of the world. When this happens, the Quest Monitor of this quest will detect an inconsistency in the quest plan (i.e. the player cannot give the antidote to his wife if he does not have it). In order to solve this inconsistency, a new plan will be requested to the Quest Planner, in an attempt to provide an alternative way to achieve the same goal of the previous plan.

In *Plan 2*, after breaking the bottle of the first antidote, the player has to go to the market and ask an old man for another antidote, get the antidote, and go back home to save his wife. However, in this new sequence of events, the event where the old man gives an antidote to the player has a precondition that indicates it can only occur if the player has been behaving in a friendly manner – which should have been true when *Plan 2* was generated. But suppose that, before asking the old man for an antidote, the player starts to behave in a more hostile manner (e.g. by not helping other characters). This behavioral change will cause the player model to be updated and the score of the Big Five factor that represents the agreeableness dimension will be decreased. When this happens, the Quest Monitor of this quest will detect this inconsistency and trigger another replanning procedure. But now the previous quest goal will no longer be achievable, because there are no more antidotes available in the game world. This will force the planner to try another authorial goal.

In the resulting plan (*Plan 3*), after trying all the alternatives to get an antidote, the only choice the player has is to go home and see his avatar John kill his wife to save her from a dreadful destiny. This new sequence of events affects the resulting world state of the quest "*Save wife*", which in turn introduces an inconsistency in the plan of its parent quest. In this situation, the quest "*Protect house*" cannot be executed anymore, because it requires the player's wife to be alive. In order to correct this inconsistency, the Quest Monitor of the parent quest will request a new plan, where the quest "*Protect House*" ends up being replaced by the quest "*Escape*".

7. Evaluation and results

In order to evaluate our approach to interactive storytelling in games, we performed three tests: (1) the evaluation of the behavior model, which aims at assessing the accuracy and performance of the Neural Network used to recognize in-game player behaviors; (2) the evaluation of the personality model, which has the purpose of verifying how well our Big Five Game Inventory replicates the results of BFI-10 in a game; and (3) the evaluation of the hierarchical quest generation algorithm to analyze its scalability and performance in highly interactive game environments. The next sections describe the methodology and the results of these tests.

7.1. Behavior model

To evaluate the player behavior model, we performed two tests: (1) a precision test to check the accuracy of the proposed model; and (2) a performance test to evaluate the real-time performance of the Neural Network used to predict the player behavior.

For the precision test, we used five datasets of different time windows to train and test our Neural Network. These datasets were created with data collected from 52 gameplay sessions (approximately 4 h of gameplay) and included samples with all the gameplay features used as input to our model, as well as the scores of the Big Five factors used to characterize the player behavior (model's output). In all the experiments, we used a 10-fold cross-validation strategy. As described in Section 4.3, each sample was analyzed and labeled by one expert. The numbers of samples of the datasets are: (1) time window of 5 s – 2903 samples; (2) time window of 10 s – 1451 samples; (3) time window of 15 s – 967 samples; (4) time window of 20 s – 720 samples; and (5) time window of 25 s – 562 samples.

Three statistical criteria were applied to evaluate the precision of our model: (1) the *root-mean-square error* (*RMSE*), which is the square root of the average squared distances between the actual score and the predicted score (prediction error); (2) the *Pearson product-moment correlation coefficient* (*r*), which measures the linear association between the actual score and the predicted score; and (3) the *coefficient of determination* (R2), which represents the proportion of the variance in the actual score that is predictable. The correlation coefficient (*r*) is represented in the interval of $[-1, +1]$. While an *r* of $+1$ indicates that the actual score and the predicted score are perfectly related, an *r* of -1 indicates that the two scores are totally unrelated. The coefficient of determination ($R^2 \in [0,1]$) can be thought of as a percentage that indicates the extent to which the scores are predictable. A higher R^2 is an indicator of better fitness for the observations. For the *RMSE* criteria, low values indicate low prediction errors.

For the precision test, we calculated the average value (10-fold cross-validation) of the evaluation criteria (*r*, R^2, and *RMSE*) for each Big Five factor obtained by Neural Networks trained with datasets of different time windows (5, 10, 15, 20 and 25 s). The average of these values for each time window is denoted by r_{BF}, ${R_{BF}}^2$, and $RMSE_{BF}$. The results (Table 3) indicate that the best length for the time window is 10 s ($r_{BF} = 0.97$, ${R_{BF}}^2 = 0.96$ and $RMSE_{BF} = 0.06$).

For the performance evaluation of our model, we tested the prediction of the player behavior during 5 gameplay sessions, wherein a total of 120 behavior predictions were performed (time window of 10 s). For each prediction, we computed the time necessary to calculate the input features and predict the Big Five factors using the Neural

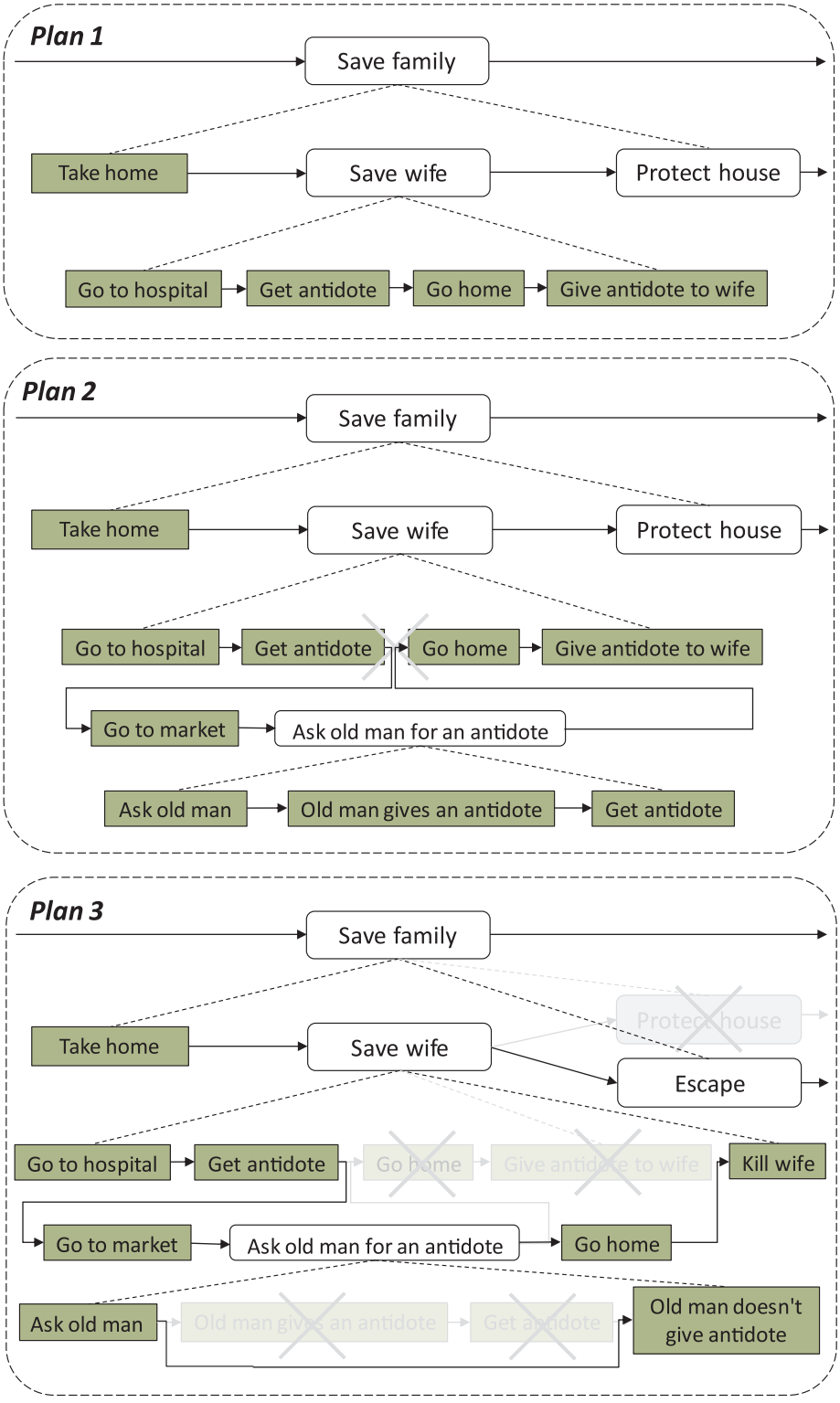

Fig. 7. Example of dynamic quest plans generated by the planner while the player is performing actions and progressing through the quest "Save family".

Network. The computer used to run the experiment was an Intel Core i7 2630QM, 2.0 GHZ CPU, 16 GB of RAM using a single core to process the Neural Network. As a result, we got an average time of 4.2 ms (standard deviation of 1.2 ms), which indicates the applicability of the proposed method in highly interactive game environments without noticeable delays.

7.2. Personality model

In order to evaluate the personality model, we performed a correlation analysis between our Big Five Game Inventory (BFGI-10) and the BFI-10 questionnaire. With this test, we aim at evaluating: (1) how well the proposed introductory scene questions represent the BFI-10 scales; (2) how well the scenes create the right situations to support the decision-making process; and (3) whether or not the players behave in the

Table 3
Average evaluation criteria (r, R^2, and *RMSE*) for the Big Five factors (Openness (O), Conscientiousness (C), Extraversion (E), Agreeableness (A), and Neuroticism (N)), obtained for different time windows, and final average values r_{BF}, ${R_{BF}}^2$, and $RMSE_{BF}$ for the precision test of the behavioral model.

Time Windows		5	10	15	20	25
r	O	0.90	0.96	0.91	0.88	0.90
	C	0.90	0.93	0.90	0.89	0.91
	E	0.93	0.98	0.97	0.80	0.77
	A	0.94	0.99	0.97	0.96	0.97
	N	0.98	0.99	0.95	0.94	0.96
	r_{BF}	0.93	0.97	0.94	0.89	0.90
R^2	O	0.81	0.93	0.89	0.95	0.81
	C	0.91	0.93	0.83	0.80	0.78
	E	0.88	0.96	0.88	0.81	0.98
	A	0.89	0.99	0.97	0.96	0.97
	N	0.79	0.98	0.96	0.90	0.87
	${R_{BF}}^2$	0.86	0.96	0.91	0.88	0.88
RMSE	O	0.09	0.08	0.10	0.15	0.13
	C	0.09	0.07	0.10	0.12	0.09
	E	0.08	0.02	0.04	0.06	0.14
	A	0.09	0.05	0.08	0.11	0.11
	N	0.12	0.06	0.08	0.14	0.06
	$RMSE_{BF}$	0.09	0.06	0.08	0.12	0.11

situations presented in the introductory game scene in ways that are consistent with their real-world tendencies.

The test was divided into three phases. In the first phase, we asked participants to freely play the game and, while interacting with the introductory scene, the game automatically recorded the players' choices for each decision-making point. In the second phase, we asked participants to fill out the BFI-10 questionnaire. In the last phase, the participants were interviewed about their experience and questioned about possible contradictions between their choices in the decision-making points and the BFI-10 questions.

A total of 36 computer science students participated in the test. Thirty-one subjects were male and five female. Ages ranged from 19 to 28 years (mean of 21.3). Twenty-eight of them play video games at least weekly.

On average, each gameplay session lasted 26.11 min, which comprise both the time players spend on the introductory scene (average of 9.4 min) and the extra time they freely spend playing the game. As expected, the traditional BFI-10 questionnaire was completed in less than a minute (average of 53.2 s).

The results of the test are shown in Fig. 8, where each bar represents the *root-mean-square error* (*RMSE*) calculated according to the participants' answers to the BFI-10 questions (observed values) and their answers to our BFGI-10 interactive questions (predicted values). Although there were some differences between participants' answers to the questionnaires, the *RMSE* is very small and for the most part results from small differences in the Likert scale (e.g. some participants that choose an alternative equivalent to "disagree strongly" in the game scene sometimes choose "disagree a little" in the BFI-10). Another very good result comes from the fact that the answers for both questionnaires matched 100% for 47.22% of the participants.

As can be observed in Fig. 8, questions 8 and 2 were the ones with higher *RMSE*. While the BFI-10 question 8 evaluates the participant as "I see myself as someone who does a thorough job", the game puts the player in a survival situation where he needs to decide how well he will fix an unsafe motorcycle before escaping from a group of zombies. In the case of question 2, the BFI-10 evaluates the participant as "I see myself as someone who is generally trusting" and the game asks the player how much he trusts an information provided by a friend of the main character. In order to understand why these questions caused confusion in some participants, we questioned them about the reasons that led them to contradictions. In the case of question 8, the participants stated that the survival and dangerous situation of the game pressed them to a more desperate decision. Similarly, for question 2 they stated that John's friend did not seem to be trustable because they (as players) knew nothing about him. Although these contradictions only occurred with 3 participants (8.33%), they indicate that special care is needed when designing introductory scenes, such as avoiding decisions that can be influenced by fantasy situations or nonexistent backstories.

In order to evaluate how much the differences between participants' answers to the BFI-10 and BFGI-10 tests affect the final scores of Big Five dimensions, we calculated the scores for both questionnaires and then computed the *RMSE* of each dimension. The results of this test are shown in Fig. 9. As can observed, the final *RMSE* is very small (< 0.09) in all dimensions, which serves to confirm the accuracy for the personality model.

We did not use any control protocol to provide sufficient personality variety in the players' sample. We simply selected a group of student volunteers, in which at least 75% of them were used to playing video games weekly and the average age was approximately 20 years. However, we checked if the results from the BFI-10 test exhibited a minimum of personality diversity from two points of view. First, we verified that the five personality traits for any pair of players were reasonably different. For this verification, we calculated the distance

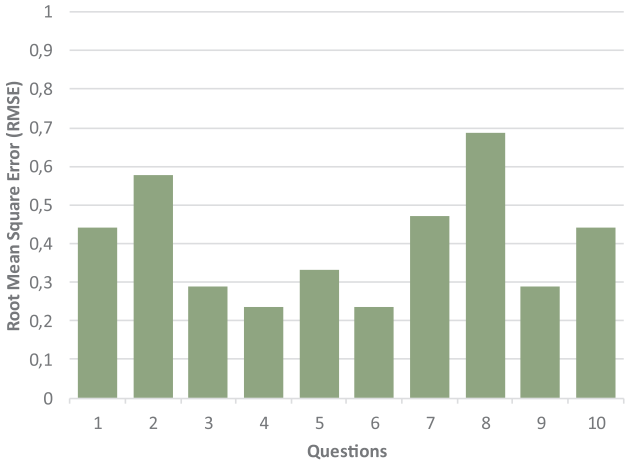

Fig. 8. Average root-mean-square error (*RMSE*) between participants' answers to the BFI-10 questionnaire (observed values) and their answers to our BFGI-10 interactive questionnaire (predicted values).

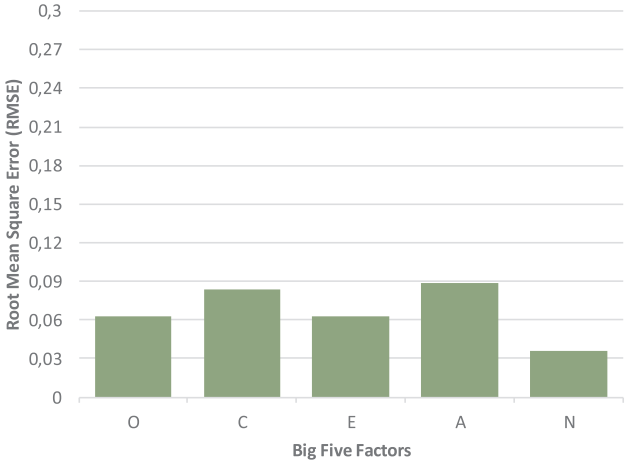

Fig. 9. Average root-mean-square error (*RMSE*) for the computed scores of the Big Five factors between the BFI-10 and BFGI-10 tests (Openness (O), Conscientiousness (C), Extraversion (E), Agreeableness (A), and Neuroticism (N)).

Table 4
Mean and coefficient of variation for the five factors (Openness (O), Conscientiousness (C), Extraversion (E), Agreeableness (A), and Neuroticism (N)).

		O	C	E	A	N
Mean	μ	0.51	0.56	0.47	0.53	0.49
Coefficient of variation	V	12%	13%	10%	13%	17%

between two sets of values as being the *root mean square difference*, that is:

$$distance(i, j) = \sqrt{\left(\sum_{k=1}^{5} (p_k^i - p_k^j)^2\right)/5},$$

where p_k^i is the k-th factor of player i and p_k^j is the k-th factor of player j. Our sample exhibited all distances between 0.10 and 0.74, which is an indication of diversity. From a second viewpoint, we calculated the coefficient of variation for each Big Five factor:

$$V_k = \sigma_k/\mu_k$$

where σ_k is the sample standard deviation and μ_k is the mean of the k-th factor observed for all players. The coefficient of variation is a measure of variability of the data around the mean, which is a dimensionless value. In our sample, V_k was between 10% and 17% around the mean, for the 36 players and the 5 factors – which is an indication of personality diversity amongst the players. Table 4 shows μ_k and V_k for the five factors and Fig. 10 shows the sets of scores for the factor with minimum value of V_k (Extraversion) and the maximum V_k value (Neuroticism). Furthermore, the means for the five factors are around 0.5, which appears to confirm the good personality variety of the sample.

7.3. Quest generation

This section presents the evaluation of the hierarchical quest generation algorithm, analyzing its scalability and performance in highly interactive game environments. This evaluation was already presented in a previous work [35], when we first developed the core of the generation algorithm.

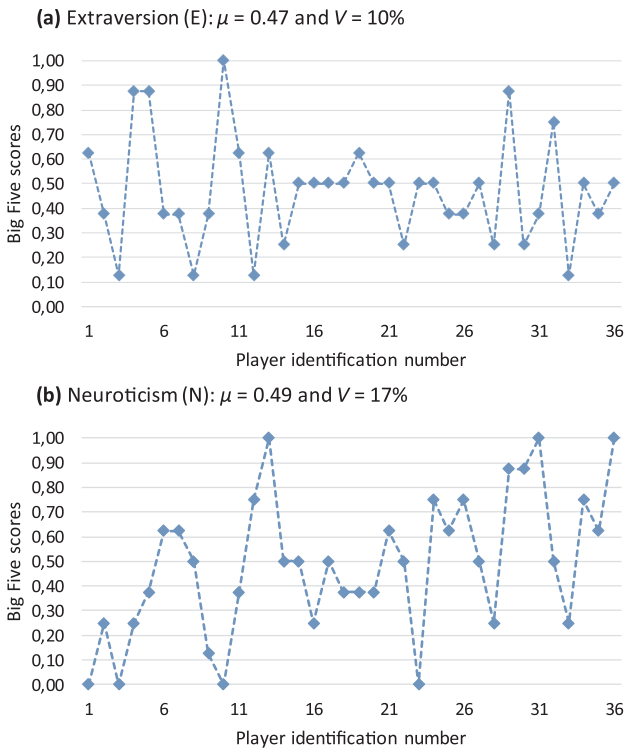

Fig. 10. (a) Extraversion scores for 36 players (minimum V value); (b) Neuroticism scores for 36 players (maximum V value).

Table 5
Solution costs for five quests with increasing number of objects (characters, locations and items) and operators. Solution cost is given by the number of events of the generated quest.

	Quest 1	Quest 2	Quest 3	Quest 4	Quest 5
Objects	12	16	20	22	36
Operators	4	6	8	9	15
Solution Cost	9	10	17	25	32

In order to evaluate the scalability of the proposed quest generation method, we conducted a performance test comparing the time necessary to generate a valid plan of actions for five quests selected from our prototype RPG with increasing levels of complexity. The planning problem was described in PDDL language and we used the HSP2 planner provided by Bonet and Geffner [52]. Table 5 shows some statistics of five selected quests, where the solution cost is given by the number of events of the generated quest ($cost(a, s) = 1$).

We created two versions of the selected quests to compare time performance. The first version was defined as a single planning problem, and the second one used the proposed approach of dividing quests into sub-quests and then creating plans incrementally as the player progresses through the quests.

The results of the performance tests are shown in Fig. 11. The left bar of each quest corresponds to the time spent by the algorithm to find a valid solution to the quest defined as a single planning problem and the right bar indicates the time required to solve the same quest problem defined as a hierarchy of sub-quests. The different colors of the right bars (indicated by numbers) correspond to the time spent by the planning algorithm to solve each sub-quest individually. The performance gain is clearly shown in Fig. 11 (in which time is presented in a logarithmic scale). The computer used to run the experiment was an Intel Xeon E5620, 2.40 GHZ CPU, 24 GB of RAM using a single core to process the planning algorithm.

Despite the replanning procedures (required whenever the player performs an action that modifies the world state unexpectedly), the efficiency of the quest planner allows the system to update the quest plans in real-time. The player performs the quests without noticeable delays.

The results of the performance experiment confirm the received wisdom that the process of generating quest plans becomes more complex as more objects and operators are added to their respective planning problems. The results also show that generating plans for complex quests using a traditional planning approach (named SP in Fig. 11) quickly become intractable (even for very fast traditional planning algorithms as the HSP2 available at [53]). In our experiment, the process to find a single plan (SP) to solve Quest 5 spent more than 12 s, which is a huge amount of time for a game that must be executed in real time. Our approach of dividing quests into sub-quests can significantly reduce the amount of processing time required to generate solutions for the quest problems. Using our method, the complete plan to solve Quest 5 was generated in 0.48 s. Our approach generates plans incrementally, as the player progresses through the game, intercalating planning and execution, which divides the planning process in multiple stages.

8. Concluding remarks

The main contribution of this paper is a new method to generate interactive and adaptive narratives for games based on the player's personality and behaviors. As discussed in Section 3, previous works on behavior and personality modeling for games adopt very simplified

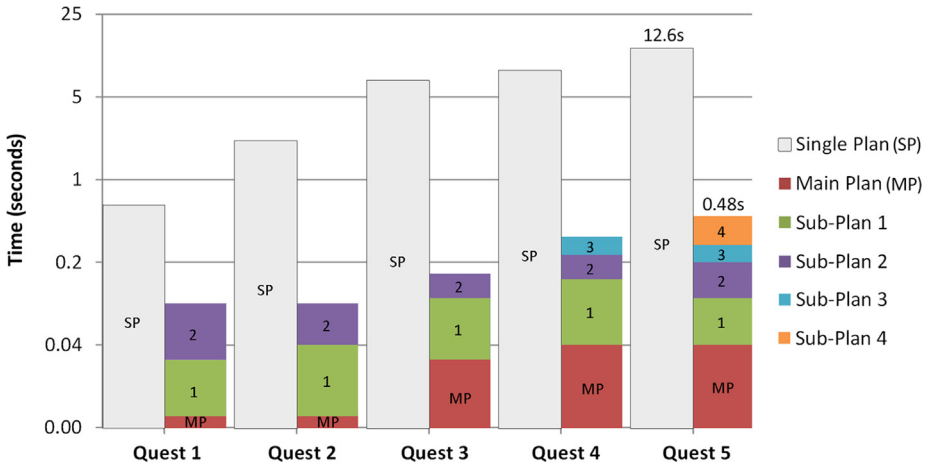

Fig. 11. Results of the performance test (time in logarithmic scale). For each quest, the left bar indicates the time spent by the planner to solve the quest as a single planning problem (SP) and the right bar represents the time spent to solve the main plan (MP) and the sub-quests in the hierarchy.

models of player archetypes, which are not adequate for representing blended behaviors and personalities. Moreover, in our approach, we maintain the rigor of the formulation of interactive narratives as a nondeterministic planning problem. As far as we are aware, no other work in nondeterministic planning deals with personality and behavior. Our method represents a more efficient and robust approach to interactive storytelling than the traditional branching storylines adopted by most video games with interactive narrative.

In practice, our method provides game designers with new ways of imagining and creating narratives for games. While the personality model allows designers to build personalized stories for specific groups of players, the behavior model allows them to prepare the game to dynamically adapt the evolving narratives according to the behavior of the players during the game. This is all supported by our hierarchical quest generation algorithm, which integrates planning, execution, and monitoring to generate and update dynamic and nondeterministic quests in real time. We believe that this form of interactive narratives can expand the boundaries of traditional games towards new forms of interactive storytelling, allowing players to create their own narrative experiences.

To support the application of our method, we developed a new Big Five inventory, especially oriented to help determining the players' personality traits, named Big Five Game Inventory (BFGI-10). This inventory, as an adaptation of the well-known BFI-10 inventory, comprises more than just questions and measurement scales. It includes 10 story-related scenes followed by decision-making points (one for each BFI-10 question), whereupon players make decisions that are equivalent to answering BFI-10 questions. Although the BFGI-10 was specifically designed for our game narrative, we believe that its scenes and questions can be easily adapted and used as a basis for assessing the personality of players in other game genres. The generalization of the Big Five Game Inventory for several types of games and the creation of tools to assist the design of BFGI-10 scenes is promising area for future research.

The results of the evaluation tests demonstrate the applicability of our method in highly interactive game environments. While the accuracy of the player models provides very precise information about the personality and behaviors of players, the efficiency of the quest planner allows the game to dynamically update quests in real time.

Although the proposed method presented good results in our experiments, some methodological considerations and current limitations of our research work must be pointed out. First, since our primary focus was on the technical aspects, we have not yet conducted a rigorous user study to evaluate our method from the player's perspective, which surely is a mandatory task that will serve as orientation for the next stages of the project. Secondly, we did not perform tests to assess the overall generalization of our method towards other game genres. In fact, some adaptations are clearly necessary when applying our method to other games, such as the definition of new relationships between behavioral aspects and in-game player behaviors (see Table 2). Thirdly, we have intentionally avoided experts in player behavior to analyze the video segments and annotate scores for the samples. We preferred to select them from a group of computer science students capable of analyzing games as experienced players. We adopted this methodology, not because player behavior experts are hard to find, but rather because we want tasks and questionnaires that can be easily completed by simply observing the player's most visible actions.

Another important remark are the difficulties that arise when applying our method to a game. First, it is important to notice that the labeling process, that is performed by experts with a minimum of knowledge about the involved game, is a very laborious and time-consuming task, mainly because the best *time window* must be found before performing the training process with the proper examples. Unfortunately, there is no way of knowing the best time window in advance, because this window depends on the overall gameplay rhythm, which is an individual characteristic of each game. The second difficulty comes from the implementation of the game environment, which must be robust enough to deal with all story variations that emerge from the hierarchical quests (such as spawning and controlling the instances of items and characters in the locations specified in quest plans, while guaranteeing the consistence between the game world and the logical state used by the planning algorithms). On this matter, the adoption of procedural content generation methods that could automatically adapt the game world to the quests is a promising direction for future research.

All in all, the positive feedback we received from players, especially the enthusiasm demonstrated by them while playing the game, is a welcome stimulus for the continuation of our work.

Acknowledgements

We would like to thank CNPq (National Council for Scientific and Technological Development), FAPERJ (Carlos Chagas Filho Research Support Foundation of the State of Rio de Janeiro) and FINEP (Brazilian Innovation Agency), which belong to the Ministry of Science, Technology and Innovation, for the financial support.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the

online version, at https://doi.org/10.1016/j.entcom.2018.08.003.

References

[1] G.N. Yannakakis, P. Spronck, D. Loiacono, E. André, Player Modeling, Dagstuhl Follow-Ups 6 (2013).
[2] S.C. Bakkes, P.H. Spronck, G. van Lankveld, Player behavioural modelling for video games, Entertain. Comput. 3 (2012) 71–79.
[3] M. Seif El-Nasr, A. Drachen, A. Canossa, Game Analytics: Maximizing the Value of Player Data, Springer-Verlag, London, England, 2013.
[4] J. Valls-Vargas, S. Ontañón, J. Zhu, Exploring Player Trace Segmentation for Dynamic Play Style Prediction, in: Proceedings of the Eleventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, Santa Cruz, United States, 2015, pp. 93–99.
[5] A. Nagle, P. Wolf, R. Riener, Towards a system of customized video game mechanics based on player personality: relating the Big Five personality traits with difficulty adaptation in a first-person shooter game, Entertain. Comput. 13 (2016) 10–24.
[6] M. Lewis, K. Dill, Game AI Appreciation, Revisited, in: S. Rabin (Ed.), Game IA Pro2: Collected Wisdom of Game AI Professionals, A K Peters/CRC Press, Boca Raton, United States, 2015, pp. 3–17.
[7] B.K. Mishra, Psychology: The Study of Human Behaviour, PHI Learning, New Delhi, India, 2008.
[8] D.N. Rapp, R.J. Gerrig, Predilections for narrative outcomes: the impact of story contexts and reader preferences, J. Mem. Lang. 54 (2006) 54–67.
[9] L.R. Goldberg, An alternative "description of personality": the Big-Five factor structure, J. Pers. Soc. Psychol. 59 (6) (1990) 1216–1229.
[10] M. Ghallab, D. Nau, P. Traverso, Automated Planning: Theory and Practice, Morgan Kaufmann Publishers, San Francisco, United States, 2004.
[11] U. Kuter, D. S. Nau, E. Reisner, R.P. Goldman, Using classical planners to solve nondeterministic planning problems, in: Proceedings of the International Conference on Automated Planning and Scheduling, Sydney, Australia, 2008, pp. 190–197.
[12] R. Bartle, Hearts, clubs, diamonds, spades: players who suit muds, J. MUD Res. 1 (1) (1996).
[13] C.M. Bateman, R. Boon, 21st Century Game Design, first ed., Charles River Media Game Development, Cengage Learning, Boston, United States, 2005.
[14] I.B. Myers, The Myers-Briggs Type Indicator Manual, The Educational Testing Service, Princeton, United States, 1962.
[15] N. Yee, Motivations for play in online games, J. CyberPsychology Behav. 9 (6) (2006) 772–775.
[16] L.E. Nacke, C. Bateman, R.L. Mandryk, BrainHex: a neurobiological gamer typology survey, Entertain. Comput. 5 (2014) 55–62.
[17] J. Tuunanen, J. Hamari, Meta-Synthesis of Player Typologies, in: Proceedings of International Nordic DiGRA 2012 Conference, Tampere, Finland, 2012.
[18] O. Missura, T. Gärtner, Player modeling for intelligent difficulty adjustment, in: Proceedings of the 12th International Conference on Discovery Science, Porto, Portugal, 2009, pp. 197–211.
[19] B. Weber M. Mateas, A data mining approach to strategy prediction, in: 2009 IEEE Symposium on Computational Intelligence in Games, Milano, Italy, 2009, pp. 140–147.
[20] T. Mahlman, A. Drachen, A. Canossa, J. Togelius, G. N. Yannakakis, Predicting Player Behavior in Tomb Raider: Underworld, in: Proceedings of the 2010 IEEE Symposium on Computational Intelligence and Games, Copenhagen, Denmark, 2010, pp. 178–185.
[21] M.C. Machado, G.L. Pappa, L. Chaimowicz, A binary classification approach for automatic preference modeling of virtual agents in civilization IV, in: Proceedings of the 2012 IEEE Conference on Computational Intelligence and Games, Granada, Spain, 2012, pp. 155–162.
[22] P.H.M. Spronck, F. den Teuling, Player modeling in civilization IV, in: Proceedings of the Sixth Artificial Intelligence and Interactive Digital Entertainment Conference, Palo Alto, United States, 2010, pp. 180–185.
[23] H. Barber, D. Kudenko, A user model for the generation of dilemma-based interactive narratives, in: Proceedings of the AIIDE 2007 Workshop on Optimizing Player Satisfaction, Stanford, California, 2007, pp. 13–18.
[24] M. Seif El-Nasr, Interaction, narrative, and drama creating an adaptive interactive narrative using performance arts theories, Interaction Stud. 8 (2) (2007) 209–240.
[25] M. Sharma, S. Ontañón, M. Mehta, A. Ram, Drama Management and Player Modeling for Interactive Fiction Games, Comput. Intell. 26 (2) (2010) 183–211.
[26] D. Thue, V. Bulitko, M. Spetch, E. Wasylishen, Interactive storytelling: a player modelling approach, in: Proceedings of the 3rd Artificial Intelligence and Interactive Digital Entertainment Conference, Stanford, United States, 2007, pp. 43–48.
[27] R. Laws, Robin's Laws of Good Game Mastering, Steve Jackson Games, Austin, United States, 2002.
[28] A. Ramirez, V. Bulitko, Automated planning and player modeling for interactive storytelling, IEEE Trans. Comput. Intell. AI Games 7 (4) (2015) 375–386.
[29] G. Van Lankveld, P.H.M. Spronck, H.J. Van den Herik, A. Arntz, Games as personality profiling tools, in: Proceedings of the 2011 IEEE Conference on Computational Intelligence in Games, Seoul, Korea, 2011, pp. 197–202.
[30] A. Bean, G. Groth-Marnat, Video gamers and personality: a five-factor model to understand game playing style, Psychol. Pop. Media Cult. 5 (1) (2016) 27–38.
[31] M. Mehroof, M.D. Griffiths, Online Gaming addiction: the role of sensation seeking, self-control, neuroticism, aggression, state anxiety, and trait anxiety, CyberPsychol., Behav., Soc. Network. 13 (3) (2010) 313–316.
[32] N.C. Worth, A.S. Book, Dimensions of video game behavior and their relationships with personality, Comput. Hum. Behav. 50 (2015) 132–140.
[33] V. Zeigler-Hill, S. Monica, The HEXACO model of personality and video game preferences, Entertain. Comput. 11 (2015) (2015) 21–26.
[34] M.C. Ashton, K. Lee, The HEXACO-60: a short measure of the major dimensions of personality, J. Pers. Assess. 91 (4) (2009) 340–345.
[35] E.S. Lima, B. Feijó, A.L. Furtado, Hierarchical generation of dynamic and non-deterministic quests in games, in: Proceedings of the 11th International Conference on Advances in Computer Entertainment Technology, Funchal, Portugal, 2014, Article N. 24.
[36] F. Strack, R. Deutsch, Reflective and impulsive determinants of social behavior, Person. Soc. Psychol. Rev. 8 (2004) 220–247.
[37] B. de Raad, M. Perugini, Big Five assessment, Hogrefe & Huber Publishers, Boston, United States, 2002.
[38] P.T. Costa, R.R. McCrae, Revised NEO personality inventory (NEO-PI-R) and NEO five-factor inventory (NEO-FFI) professional manual, Psychol. Assess. Resour. (1992).
[39] S.D. Gosling, P.J. Rentfrow, W.B. Swann, A very brief measure of the Big-Five personality domains, J. Res. Pers. 37 (6) (2003) 504–528.
[40] M.D. Back, S.C. Schmukle, B. Egloff, Predicting actual behavior from the explicit and implicit self-concept of personality, J. Pers. Soc. Psychol. 97 (3) (2009) 533–548.
[41] S.V. Paunonen, Big five factors of personality and replicated predictions of behavior, J. Pers. Soc. Psychol. 84 (2003) 411–422.
[42] H. Borchani, G. Varando, C. Bielza, P. Larrañaga, A survey on multi-output regression, Wiley Interdiscipl. Rev. Data Min. Knowl. Disc. 5 (5) (2015) 216–233.
[43] S. Haykin, Neural Networks and Learning Machines, Prentice Hall, Upper Saddle River, United States, 2008.
[44] R. McCrae, P. Costa, Personality trait structure as a human universal, Am. Psychol. 52 (5) (1997) 509–516.
[45] M. Back, B. Egloff, Yes we can! A plea for direct behavioral observation in personality research, Eur. J. Person. 23 (2009) 403–405.
[46] N. Peever, D. Johnson, J. Gardner, Personality & video game genre preferences, in: Proceedings of The 8th Australasian Conference on Interactive Entertainment, Auckland, New Zealand, 2012, Article No. 20.
[47] S.H. Hsu, C.H. Kao, M.C. Wu, Factors influencing player preferences for heroic roles in role-playing games, Cyberpsychol. Behav. 10 (2) (2007) 293–295.
[48] B. Rammstedt, O.P. Johnb, Measuring personality in one minute or less: a 10-item short version of the Big Five Inventory in English and German, J. Res. Pers. 41 (1) (2007) 203–212.
[49] S.D. Gosling, P.J. Rentfrow, W.B. Swann Jr, A very brief measure of the Big-Five personality domains, J. Res. Pers. 37 (2003) 504–528.
[50] G. Guido, A.M. Peluso, M. Capestro, M. Miglietta, An Italian version of the 10-item Big Five Inventory: an application to hedonic and utilitarian shopping values, Person. Individ. Differ. 76 (2015) 135–140.
[51] R. Carciofo, J. Yang, N. Song, F. Du, K. Zhang, Psychometric evaluation of Chinese-Language 44-Item and 10-Item Big Five personality inventories, including correlations with chronotype, mindfulness and mind wandering, PLoS One 11 (2) (2016) 1–26.
[52] B. Bonet, H. Geffner, Planning as heuristic search, Artif. Intell. 129 (1) (2001) 5–33.
[53] HSP-Planners, The HSP family of planners. https://code.google.com/p/hsp-planners/ (accessed 27 February 2018).