MENTA: how to balance authorial intention and user agency in virtual environments

Domitile Lourdeaux, Mohamed Sallak, Rémi Lacaze-Labadie

MENTA: how to balance authorial intention and user agency in virtual environments

1st Domitile Lourdeaux
Alliance Sorbonne université
Université de technologie de Compiègne
CNRS, Heudiasyc UMR 7253
Compiègne, France
domitile.lourdeaux@hds.utc.fr

2nd Mohamed Sallak
Alliance Sorbonne université
Université de technologie de Compiègne
CNRS, Heudiasyc UMR 7253
Compiègne, France
mohamed.sallak@hds.utc.fr

3rd Rémi Lacaze-Labadie
Alliance Sorbonne université
Université de technologie de Compiègne
CNRS, Heudiasyc UMR 7253
Compiègne, France
remi.lacaze-labadie@hds.utc.fr

***Abstract*—We aim to create an artificial intelligence based virtual environment to train medical team leaders to rescue injured people after a mass casualty. In this paper, we describe a resilient and adaptive engine, MENTA, to orchestrate dynamically various training situations and various virtual teammates. MENTA is in charge of the narrative control by proposing a set of adjustments that satisfies narrative objectives chosen by the trainer. These adjustments take the form of a prescribed scenario that is generated by MENTA via a planning engine that we have coupled with fuzzy cognitive maps. This approach tackles the problem of a set of objectives that are often contradictory: user agency, authorial intention and resilience.**

***Index Terms*—Virtual Environments, Planning, Fuzzy Cognitive Map**

I. Introduction

In the VICTEAMS project, we aim to create a **virtual environment** to **train medical team leaders** to rescue injured people during a mass casualties situation (e.g. military fighting, terrorist attacks). Such situations are crisis situations: when they occur, they are often unexpected and they require rescue teams actors to adapt quickly and make difficult choices. The team leader has to be trained how to be resilient under high emotional pressure. To make such training through personal experience easier, we want to confront the medical leader with **various contexts and teammates**. To support this kind of training, a huge authoring work is needed. This amount of work to scale up to coherent and precisely controlled scenarios is called authoring bottleneck [1]. It shows the necessity to set up an orchestration system, able to generate automatically adaptive environments, without having to define specifically every possible scenario. This paper focuses on the **dynamic orchestration process of these situations in a virtual environment**. Orchestration is a process combining both the specification of the possible or desirable progress(es) of the simulation and the control (performance, monitoring, correction) of the events progress in interactive time. We present a software platform as an orchestration system aiming towards apparently contradictory objectives:

- **user agency**, i.e. the possibility for the user to act as freely as possible in the environment, which is required for constructivism training,
- **authorial intent** to guarantee interesting situations, to control them and to encourage learning,
- **resilience** to provide a dynamic adaptive system, able of running a scenario which meets training objectives despite the uncertainty in the user's or characters' actions,
- system **adaptability**, essential to provide scenario **variability** and the **scalability** of the simulation with the emergence of new situations.

The scientific issue is created by the reaching of these incompatible objectives. The addressed research question is: **how can we dynamically generate relevant and personalized crisis scenarios that satisfy all these objectives?**

To tackle this issue and to generate these various situations, we propose a **compromise between a scripted approach and a dynamic generative approach**. Our approach is based on **autonomous virtual characters** and an **dynamic orchestration engine**. The characters play the role of the medical teammates and must have natural interactions with each other. Although the trainee can act directly on the environment and on the victims, she/he has to learn to manage her/his virtual team, to delegate actions, to manage her/his stress but also her/his team, to supervise and to collect information.

Our approach is based on planning to predict both trainee and autonomous character behaviours, and to generate a set of adjustments that would indirectly influence the character behaviours, guiding the scenario towards desired situations. There are many hybrid systems which attempt to reach a compromise between oriented scenario approach and emergent approach. But most of them do not consider the complexity of the crisis situation, the human factors aspects and the uncertainty of the situation evolution. Our contribution aims to propose a generative powerful and a high resilience to tailor and to control crisis situations in interactive time and in uncertain context.

In this paper, we start by presenting closely related work that deals with virtual environment for crisis management training and orchestration systems. This is followed with an overview of our general approach. Then we describe our model and the planning. We finish an evaluation of our approach.

II. Related works

Two main approaches are often opposed: **plot-based approaches and character-based approaches**. **Plot-based approaches** such as [2] focus on the overall quality of the scenario. This approach, however, suffers high computational costs to ensure both scalability and user agency. Moreover, a scenario controller dynamically modifies the characters features, but sometimes without coherence between their state of mind and what is happening in the environment. [3] have yet shown that incoherences in the character motivations have a negative impact on the user experience and understanding. If there is a great deal of change in character behavior, then the user may become confused or disinterested in the resulting inconsistent character behavior [4]. Moreover, in a training context, it makes it difficult to have explainable behaviours for reflexive learning. It is difficult to ensure the coherence of the behaviours, and to find explainable behaviours. Some systems try to address this problem of characters goals coherence. But it is still an open area of research and is not performed in a narrative-theoretic manner. As an example, [5] proposes plan distance metrics and intentional plans to be aware of a trajectory set, then one could be chosen that ensured desired outcomes, such as one that minimized changes to character goals. With **character-based approaches** such as I-Storytelling [6] or EmoEmma [7], the scenario emerges from the interactions between the user and the virtual characters. Control is shared, as every character is responsible for its decision making. These approaches focus on creating virtual characters with a high cognitive decision process. While these characters are dynamically adaptable and offer a great user agency, controlling the scenario remains difficult. Moreover, these approaches fail to achieve qualitative pedagogical objectives.

There exist many **hybrid systems** with semi-autonomous characters presenting pseudo-cognitive behaviours such as Scenario Adaptor [8], ISAT [2], IN-TALE [9]. Similarly, starting from a character-based approach, FearNot! [10] and MRE [11] offer orchestration systems integrated in the characters cognitive model. Two systems should be highlighted. [12] proposes a very sophisticated social planning with motivation control. Thespian proposes to launch a computation system at the initialisation for the virtual characters features and for events planning [3]. However [12] requires a very specific and binding definition of macro-actions, while Thespian is not resilient and does not allow to adjust the scenario dynamically or to redirect the scenario in interactive time.

To avoid this **authoring bottleneck**, systems combining control and adaptability frequently only apply an extra layer of control on a simulation made of independent entities. It is also common that those system interventions interfere with the coherence by modifying on the fly states of the simulation.

Another opposition is often proposed: **scripted approaches** and **generative approaches** [13]. **Scripted approaches**, often based on behaviour trees, allow to ensure the authorial intent, but the user agency and the adaptability are limited. On the contrary, a generative system allows to generate a wide variety of scenarios and to regenerate new scenario on the fly when the trainee goes against the prescribed scenario. **Generative approaches** based on planning allow a high resilience. However, the more generative a system is, the more difficult it is to produce a scenario which is consistent for the original authoring. The generative systems require enormously large gameplay datasets. Today, works like [14] generate policies for interactive narrative planners with deep learning. These deep learning based systems needs many electronic data, and it was impossible to collect so much data in the VICTEAMS project. Another problem with a machine learning approach, is the difficulty to reason on knowledge and the lack of explainable power. Indeed, in training session, the orchestration engine must propose adjustments to the trainer (either when the learner jeopardizes the prescribed scenario, or when the trainer wishes to modify elements). The orchestration system must be able to explain its choices to allow the trainer to better understand the impact of certain actions on the Artificial Intelligence. At last, the most of generative processes are created off-line and are not dynamic processes.

III. MENTA, Overview and positionning

To provide resilience and explainability, we propose MENTA, a hybrid approach: a **character-based and generative approach based on a dynamic orchestration and autonomous virtual characters**. The scenario experienced by the trainee will emerge from the interactions between the trainee and our orchestration system. The autonomous characters allow to provide resilience, coherence and a better user agency. To control the scenario, we consider orchestration as an extrinsic additional step for framing an existing virtual environment and not as part of the environment design process. Orchestration consists here in narrowing the space of the possible scenarios for the simulation, targeting a specific use. **The originality of our approach is a resilient generative system thanks to the couple of: 1) a numerical approach with probabilistic graphical models to respect the authorial intention and to manage uncertain contexts and 2) a symbolic approach with planning techniques to provide resilience and scalability**. MENTA provides a **compromise between a scripted approach which can be instantiated according to the context and a dynamic generative approach**. The graphical model allows to describe only **short sequences**. The purpose is not, as would do a state chart or a behviour tree, to give a complete description of the possible changes of the state of the world, but **to focus on what is relevant** for an application in particular according to the **authorial intent**. These sequences are designed by the experts, which ensure/guarantee the coherence and the pedagogical relevance. They can be designed by our editor (with drag and drop of items).

MENTA receives pedagogical objectives of the train or of a pedagogical module or new information on the world state, it creates a planning problem with this goal and the current world state. It uses this problem and the previously generated

planning domain to plan a new scenario. The consistency of the system is checked by the precondition and postcondition of the short sequences. Then, MENTA monitors the execution of actions and behaviours and triggers adjustments in the world state or on virtual characters behaviours. To influence the simulation without modifying defined objects states or giving orders to the virtual characters, MENTA can request three types of adjustments:

1) **Happenings** are exogenous events that can be triggered with no effect on the system consistency (e.g explosion).
2) **Late commitments** enable to define progressively during the simulation states left uncertain at initialisation (e.g. add a hidden injury).
3) **Occurrence constraints** ensure virtual characters coherence and by doing so the explainability of their behaviours, while providing them with a wide range of behaviours, we focused on modelling autonomous virtual characters with representative cognitive behaviours, especially erroneous and collective behaviours can be modified to override random choices in probabilistic behaviours (e.g. “apply an intravenous” has a fail probability of 40%, it is possible to force it at 0 or 100%).

Adjustments are not triggered when the scenario is generated, but when needed, for the situation to stay open as long as possible. The resulting plan is not prescriptive, but serves as a prediction of the simulation behaviour.

IV. Probabilistic graphical model

We assume that without any **interactive time pedagogical control**, the efficiency of the training is not guaranteed. The simulation could go in any direction, yet we would want it to be relevant to the profile and current state of the trainee, to the trainer objectives or to the authorial intent. Without control, it is impossible to ensure a gradual learning process, whether it concerns the abilities and the skills the trainee has to demonstrate, the difficulty/complexity of the situation, or the severity of the consequences of her/his errors.

Few models allow to describe complex crisis situations for generative simulations through expert knowledge [15]. But, these models cannot integrate **uncertainty parameters**. These **uncertainty parameters** can be expressed quantitatively, i.e. by the probability, frequency and/or occurrence rate of a critical event. It can also be expressed qualitatively, using terms such as ”often”, ”sometimes”, ”rarely”, etc. We want to generate crisis situations for which we are not able to quantify everything. We are looking to describe causal chains on **qualitative data**, that is described using **linguistic terms**. *For example, experts cannot tell that a medical operation might fail if the carer is under 10 percent stress, they would rather say that it might fail if the carer is highly stressed.*

Modeling this expert knowledge and these parameters directly by current planning languages generates approximations that can affect the overall consistency of the scenario. We assume to use graphical models, and in particular **Fuzzy Cognitive Maps** (FCMs) which are a combination of fuzzy logic and cognitive mapping in order to describe these sequences. FCMs are **intelligible** for experts and **interpretable**, i.e. interpreted by computer systems. FCMs allow **fuzzy reasoning** and thus to **reason about linguistic terms**.

We therefore took advantage of the fuzzy reasoning proposed by the FCMs to define a new type of preconditions, called fuzzy preconditions, which make it possible to model expert knowledge with natural language and take into account the uncertain values that we have to deal with. FCMs are directed graphs, used in decision theory to model complex systems and expert knowledge [16]. FCMs enable to connect events using subsumption or causality relationship through logic gates AND/OR.

FCMs allow to define **fuzzy preconditions** to provide **explainability and variability**. Usually, PDDL or STRIP planners support **hard preconditions**. Hard preconditions are Boolean and imperative preconditions (which must be satisfied) which are defined by a set of predicates linked by conjunction, disjunction or negation relationships (e.g., (A and B) or (C and not D)). These preconditions are often too rigid to model subtleties of real problems. This observation has led to the emergence of **flexible planning** [17], which is a kind of planning aims to make the preconditions of operators more flexible. To illustrate the problem, let’s take an example related to our application framework. On the figure 1, we define the preconditions of an occurrence constraint change *(e.g. ”To prepare medication” fails)*. The orchestration system can want to force the occurence constraint to create difficulty or to deal with skills as in the piece of scenario the Fragment 1 in the figure 3. To provide explainability and to prevent scenario from incoherences as shown by [3], our orchestration system do not randomly force the result of the action without scenaristic coherence. The trainee must understand why some tasks fail to provide her/him an opportunity to react. The preconditions of the success/fail of the task are the stress and the experience of the carer. Only under certain conditions mean a medication preparation fail *(e.g., if the carer is ”very stressed and experienced” or ”low stressed and unskilled”)*. With hard preconditions, it is not possible to define correctly this constraint occurrence change. Indeed, the success or the failure does not depend particularly on one factor but rather on a pondered function of these factors. If a scenario with a difficulty level of 1 is asked and the failure of a medication preparation is marked as difficulty 1, then the planner will seek to satisfy the preconditions that will lead to lead to the task failure (*e.g., by having a new victim, which will increase the carer’s stress and therefore the probability that she/he will fail the medication preparation or by triggering a happening ”stressful event”)* (see the Fragment 1 in figure 3, the carer can be played by an Autonomous Virtual Character - AVC). In addition to failing to express these nuances with hard preconditions, it is difficult for experts to give thresholds for exact values. It is more natural to use linguistic terms, for example ”if the stress is high then ...” rather than numerical values like: ”if the stress is greater than 8 then ...”.

Existing solutions for proposing flexible planning are particularly interested in **soft preconditions** which are optional

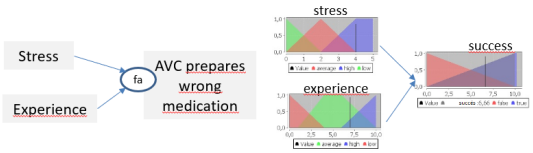

Fig. 1. Fuzzy preconditions example (fa: activation function)

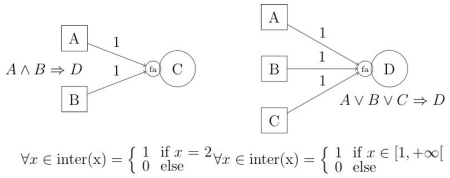

Fig. 2. Examples of AND/OR combined relations with their interval functions

preconditions which increase the overall quality of the solution if they are satisfied [17]. These works mainly solve the problems in which there are several preconditions which cannot all be satisfied. They propose to define a degree of satisfaction for each precondition, then to satisfy the preconditions at best by maximizing the sum of the satisfaction degrees. Here, we do not seek to relax certain preconditions but rather to define complex relationships between the preconditions and especially to reason on linguistic terms rather than numerical values. This is why **we have proposed fuzzy preconditions** which are more flexible than hard and classical soft preconditions.

Each FCM **node** is either associated with a **narrative operator**, in which case we call it **operator node** *(e.g., arrival of a victim)*, or with a **narrative predicate** which we call **state node** *(e.g., state victim's health)*. The state nodes can be propositional state nodes if they are associated with predicates of the proposition type or numerical state nodes for those associated with numerical predicates. State nodes act as system inputs, that is, they are used to initialize the FCM from the current state and will not change value during the execution of the FCM. Conversely, the operator nodes are the outputs and are regulated by an **activation function** to take only a value of 0 or 1. An operator node is said to be activated if its output value after execution of the FCM is equal to 1 and deactivated if it is equal to 0.

In the FCMs, an **arc** between two operator nodes is **a temporal or causal relationship**, that is, the source node must occur before the destination node. A temporal relation is for example *"to take tourniquet -¿ to apply a tourniquet"* and a causal relation is for example *"to apply a tourniquet -¿ stop of the bleeding"*. An arc from a state node to an operator node is a precondition relationship, that is, the state node is a precondition on the activation of the operator node. We can see in figure 1 that arcs do not arrive directly on a destination node but on an activation function. In addition to regulating the output value of the nodes, the **activation functions** make it possible to define **different types of relationships**. One of the reasons that FCMs are particularly expressive is the ability to define their own relationships.

With FCMs, we can define **non-fuzzy preconditions**. In this kind of precondition, we distinguish direct relations from combined AND / OR relations. Direct relationships are relationships that involve only two nodes (eg, "if A then B"). The source node A can be a propositional state node or an operator node while the destination node B is necessarily an operator node. State nodes are never destination nodes because they act as inputs to the model. The activation function used for this type of relationship is a sign function which for example in the case of "if A then B" returns 1 if the input value is positive and 0 otherwise. Combined AND/OR relationships involve multiple source nodes and a destination node. They make it possible to define conjunctions or disjunctions between several source nodes (e.g., "if A and B then C", "if A or B or C then D"). Source nodes cannot be digital state nodes because the conjunction or disjunction is based on Boolean logic. On the other hand, with fuzzy relations it is possible to use digital state nodes in AND/OR relations. The activation function used for combined AND/OR relationships is an interval function. This function can be configured to obtain an AND, an exclusive XOR or an inclusive OR. It returns 1 (true) or 0 (false) depending on the value of the source nodes and the type of relationship. In the case of AND, the interval function $inter(x)$ is equal to 1 if x is equal to the sum of the incoming nodes and 0 otherwise (we recall that the value of the source nodes is regulated to take only a value of 0 or 1). For the exclusive XOR, $inter(x)$ is equal to 1 for x equal to 1 and 0 otherwise. Finally for the inclusive OR, the function is 0 on $]-\infty, 1[$ and 1 on $[1, +\infty[$. Examples of combined AND/OR relationships are given in the figure 2. Thanks to these fuzzy and non-fuzzy preconditions, FCMs are **expressive**. They allow to represent different pedagogical, narrative, human factors or crisis features (skills, difficulty, severity, or any feature trainers need).

FCM are used to specify the causal links between the **events of interest** that can occur in the virtual environment (risk causal chains, pedagogical causal chains or narrative causal chains). We have chosen to work sequences of interest like the micro-non-technical skills/abilities. For example, leadership or situation awareness are too abstract and trainers prefer describe abilities like "to manage teammate's misunderstanding". We notice them NTS (figure 3). With FCMs, authors can describe fragments of scenario in the form of small scripted sequences of actions/events. We call these models FRAGs. Two kinds of **fragments** are described in (figure 3). Each fragment addresses a short sequence to work on a micro-non-technical skill. FCMs facilitate the **modelling of scenaristic content** as well as the **authorial intent** thanks to **fragments of scenario** that are scripted by the author and used during the planning process. Our orchestration engine uses these FRAGs, as macro-operators, to generate scenarios. However, the planning operators resulting from those models are not generated upstream,

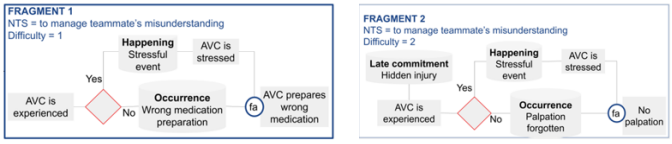

Fig. 3. MENTA process

but directly during the planning process, as their generation depends on the state of the world. The fragments are scripted but not instantiated. The planner will instantiate the fragment according to the context (e.g. the victim, injury, medication choices, the most relevant autonomous virtual carer).

V. Planning based orchestration engine

In order to control training situations, and to maintain the world consistency, MENTA, guides indirectly the events progress, by adjusting occasionally the state of the world without MENTA giving instructions to the characters. MENTA is based on a **planner** which generates a scenario according to suggested pedagogical objectives and monitors its implementation in interactive time.

The **generated scenario** by MENTA is **hypothetical**, that is to say that it is not supposed to be executed. It contains **adjustment operators** (commitments, happening and occurrence constraints). Adjustement operators are executed. But it also contains **operators of predictions** (the supposed actions of the virtual characters and of the trainee) that may finally not occur. MENTA generates a partial scenario (i.e. a scenario which does not tell the whole story but only pieces of the story to orient the trainee toward training situations of interest). Our approach makes it possible to carry out a control while offering a great user agency. However, this requires the establishment of a **resilient system** that can **regenerate** a new scenario from any state in the world. The system has to consider the scenario deviations due to the actions of the trainee or of virtual characters or a trainer modification that can go against scenario prescribed. That's why MENTA relies on a planning-based approach to generate its scenarios. **Planning** is recognised for its **generative power** and **exploration capabilities**, which allows not only to provide a scalable approach to generate a wide variety of scenarios (variability and scalability) but also to easily regenerate new scenarios if the initial scenario is no longer possible (the resilience). However, the weak points of these approaches are a difficult modelling of the script content [18] and a weak authorial intention [19]. Research works have shown that facilitating the appearance of specific operator sequences during the scenario generation process increases the author's intent [20], [21].

The generated scenario is a set of adjustments. These adjustments take the form of a prescribed scenario generated via a **planning engine coupled with fuzzy cognitive maps** through a macro-operator FRAG (figure 3). FRAG is used to model **FRAGment of the scenario** in the form of scripted sequences of actions/events. Then fragments of the scenario scripted by the author are used during the planning process.

Classic planning is a process of finding a solution for a given problem in a space of potential solutions. The nodes represent actions (in the sense of planning) and the potential solutions are all the paths going from "start" to "end". Unlike scripted scenarios, scenarios are not known in advance but exist virtually in the action space. The planning process explores this space for action, discovers potential solutions and selects those that best meet the script objectives. However, keep in mind that: (1) very often this process does not explore all the solutions; (2) that it is guided by a heuristic that directs research towards the best solutions; and (3) that there may also be no solution. The specific planning we propose has the specificity of searching in the action space and FRAGs. It adds static links in the planning process (in addition to virtual links). These static links come from the FCM's causal relationships and are known in advance because they are explicitly defined by the author. It is this coupling between static links and virtual links which makes it possible to make the compromise between the generative power of the scripting system and the respect for the intention of the author. On the one hand, these static links reinforce the intention of the author because it facilitates the emergence in the scenario of certain sequences of actions explicitly defined by the author (those linked by static links). On the other, static links add constraints to the planning process, which lowers the generative power of the system. In other words, by modifying the granularity of the system, i.e. the size of the FRAGs, we can vary the number of static links and therefore adjust the generative power of the system and strengthen / reduce the intention of the author.

The scenario is partial and prescribed, it will therefore have to evolve during its execution or even be entirely regenerated. The planner takes as input the planning area, the initial state and the scripting objectives and generates a scenario as output. This scenario is passed to the execution engine which is responsible for executing it in the virtual environment. We recall that the scenario is made up of adjustment operators (executed by the execution engine) and prediction operators (observed by the execution engine). An adjustment operator is executed in the virtual environment when all of its predecessors are finished. For prediction operators, the execution engine waits to be notified by the virtual environment of their observations and notes them as "finished" if necessary. During the execution of the scenario, an operator will be said to be on the border of the scenario if all of its predecessors are marked as finished, this means that the operator is ready to be executed or observed. The execution engine is also responsible for detecting inconsistencies in the plan and replanning if necessary. An inconsistency is detected if the preconditions of a scenario border adjustment operator are no longer met in the current state of the world. This can happen if for example the state of the world has changed unexpectedly due to a task of a virtual character or of the trainee. As for inconsistencies in observable operators, it is more difficult because it is difficult to know when we can assume that a character will not do the task expected of her/him. In some cases, we can reason about the state of the world to determine that a character's

task is no longer achievable, if for example, a character must pose a tourniquet but there are no more available. However, this reasoning is not always possible, if for example there are always tourniquets available but the character has decided not to pose a tourniquet for another reason. In this case, the prediction operators have a ttl (time to live) which indicates that after a certain delay they are considered inconsistent. Once an inconsistency is detected, the execution engine asks the planner to reschedule a new scenario from the state of the current world. In this case, the objectives may have changed if the trainer asked for it.

VI. Algorithms

In this section, we present our algorithms: 1) the processing algorithm of a FRAG operator responsible for inferring the situations that are applicable for a given state of the world, 2) the scenario generation algorithm and 3) the algorithm that transforms a temporal plan into a partially ordered plan.

A. FRAG Operator Processing

The classical planning process explores the state space by iteratively applying the applicable actions in the state being explored. This process starts with the initial state, then chooses the next state to explore according to an evaluation function until it encounters a goal state. In our case, we also plan in the space of FRAGs, which makes the planning process different because a FRAG is not directly applicable in a state. Indeed, a FRAG models a set of situations, some of which will be applicable in the state and others will not. It is therefore necessary to go through an intermediate step which must first infer the situations applicable in the state. This is what we call the processing of a FRAG, which consists of inferring the applicable situations for a given state and generating sequences of actions that can then be applied in this state. The algorithm for processing a FRAG consists of three steps: (1) the initialization of the FCM from the state of the world, (2) the execution of the FCM and (3) the generation of applicable action sequences. The execution of a FCM consists in iteratively updating the value of each node according to the nodes which influence it, and this for a fixed initial condition.

B. Scenario generation algorithm

The main difficulty is to do research in two spaces (the space of actions and the space of FRAGs) while planning is usually done in a single space (the space of actions). We use a forward chaining algorithm. At each iteration, the algorithm chooses the best state to explore according to the evaluation function $f(n)$ (defined in past section), then it explores it by applying all possible actions and processing all FRAGs applicable before inserting the resulting action sequences into the plan. The Algorithm 1 takes as input the initial state, the goal to be achieved, the set of instantiated operators (actions and FRAGs) and the list of desired NTS. We start by defining frontier as the set of nodes ready to be explored and explored as the set of nodes already explored. In line 2, $frontier$ is initialized with the first node itself consisting of an empty plane and the initial state s0. In line 3, $explored$ is initialized to empty. The algorithm runs as long as $frontier$ is not empty and no solution has been found. At each iteration, the most promising node is selected by minimizing the function $f(n)$ (line 5). In our case, minimizing the function $f(n)$ means looking for a solution that best satisfies the training criteria and minimizing the size and duration of the scenario according to the values of α and β. $frontier$ and $explored$ are then updated (line 6) then node n is explored unless it satisfies the goal of problem g, in which case the $BuildPOPPlan$ function is called. This function, which we detail in the next section, transforms a temporal plan into a partially ordered plan. The expansion of node n is done by applying to it all the actions applicable in s (line 9) and by processing then applying all the applicable FRAGs (lines 10 to 17). To do this, all applicable FRAGs are selected (line 10), then each of these FRAGs is processed by FRAG Operator Processing Algorithm (line 12). Finally, each resulting sequence of actions is applied in s if at least one of the actions of the sequence involves one of the NTS required in the problem at the level of sk^*. Applying a sequence of actions in s consists in applying each action one after the other starting from the state s. In line 18, the pruning step aims to get rid of all unpromising nodes, namely all nodes whose state s has been reached by a better plan. Finally, $frontier$ is updated with the new nodes reached (line 19) then this process is repeated until a state satisfying the goals of the problem is reached or $explored$ is empty.

Our algorithm is guided by a search heuristic that tends towards optimality but it does not guarantee it. For the algorithm, optimality is defined by the minimization of the evaluation function $f(n)$, so it is a solution that best satisfies the criteria training and which is the shortest in terms of time and number of actions. We are not looking for an optimal solution to several reasons: (1) solving an optimal problem is much more complex and time-consuming than a satisfaction problem, (2) optimality is not a strong constraint in our work because it does not guarantee a solution of quality (because it is very difficult to encode the quality of a scenario in a function) and finally (3) in the context of situated learning, we prefer to obtain several non-optimal solutions, even if they do not perfectly satisfy the scenario objectives, rather than a single optimal solution. To provide this variability, it is possible to configure the algorithm so that it stops after finding n solutions rather than just one and then randomly selects a solution or lets the trainer select the one she/he prefers.

C. Construction of the partially ordered plan

The purpose of temporal planning is to produce a temporal plan (TPlan) so that the total execution time is minimal. The consequence of this is that the temporal constraints are sufficient for each action at a time t to have its preconditions satisfied, since each action will be executed without delay. In the case of scenario generation, the total execution time is not minimal because of the observable actions (resulting from the prediction operators) whose time t is unknown since it depends on the trainee or the virtual characters. It is therefore necessary

Algorithm 1 Scenario generation

function GENERATESCENARIO(s0, g , A, FRAGS, sk*)
$frontier \leftarrow (<>, s0)$
$explored \leftarrow \varnothing$
while $frontier \neq \varnothing$ **do**
  $selectn \leftarrow (\pi_s^t, s) \in frontier$
  $frontier.remove(n), explored.add(n)$
  if $s \vDash g$ **then**
    return $BuildPOPPlan(\pi_s^t, s0)$
  end if
  $children \leftarrow (\pi_s^t.a, \gamma(s,a) | s \vDash pre_a)$
  $frags \leftarrow (frag \in frags | s \vDash pre_{frag})$
  for each $frag \in frags$ **do**
    $list\_seq\_a \leftarrow$ PROCESS$FRAG(frag, s)$
    for each $seq\_a \in list\_seq\_a | \exists a \in seq\_a, sk_a \cap sk* \neq \varnothing$ **do**
      $S' \leftarrow S$
      for each $a \in seq\_a$ **do**
        $S' \leftarrow \gamma(s', a)$
        $children.add(\pi_s^t.a, s')$
      end for
    end for
  end for
  $prune(frontier \cup explored \cup children)$
  $frontier \leftarrow frontier \cup children$
end while
end function=0

to transform the temporal plan into a partially ordered plan (POPlan) defined by causal constraints.

The algorithm builds a POPlan from a TPlan. It takes as input a TPlan (*tplan*) as well as the initial state (*s0*) and returns a POPlan (*poplan*). The general idea is to start from the TPlan, execute it action by action and fix it as you go to form the POPlan. Repairing the plan consists in transforming the temporal relations into causal relations (by modifying the successors and the predecessors of each action) according to the dependencies between the actions. We will say that an action A depends on an action B if the effects of A are necessary to satisfy the preconditions of B.

D. Example

The trainer chooses to work the micro-non-technical skill "to manage teammate's misunderstanding" with a difficulty level 1. The planner tries to apply the FRAG 1 in a partially-ordered plan. This fragment involves a victim who needs a medication. Then, it calculates that a novice carer which administrates a wrong medication is more explainable. It inserts the fragment in the planning process and predict the novice nurse "Stéphane" will take care of a victim who need medication. It plans to change the occurrence constraint of the action "to prepare medication" in order to fail it. The plan can be endangered. For example, if the leader orders to the experienced nurse "Christian" to take care of the victim. Then, the planner changes the plan. It plans to trigger a stressful happening in due course to stress him.

VII. PERFORMANCE TEST

Our performance tests depend on the the number of nodes explored during the planning and to the planning time. They were made on problems arising from international planning competitions for the following reasons: (1) it allows to have a point of comparison with classical planning problems, (2) the planning domains are available online and the problems are complex enough to test the performance of our system, (3) it shows that our approach is generic and finally (4) we believe that this could open perspectives for the use of FRAGs on problems other than narrative problems. These tests were performed on a machine with a 2.7GHz Intel i7-6820HK CPU and a 4 GB memory limit for the JVM. First, we tested the difference in terms of planning time and nodes explored between planning in the space of actions and FRAGs and classic planning (which is done only in the space of actions). Then in a second step, we tested the impact on the total planning time of the treatment of FRAGs. These two tests were performed on two different planning domains.

A. Comparison test with and without the use of FRAGs

Planning with FRAGs generates an overhead because it involves running FCMs and generating action sequences. This is why we assess the impact in terms of performance of their use. We chose the Ze-noTravel[1] planning domain which consists of picking up and disembarking passengers at different locations. This problem includes the possibility of traveling more or less quickly but we consider in our tests only one speed. We performed our tests on two models of the domain using FRAGs in addition to the basic model which does not contain any. We compared the execution time of these three models on identical problems. For the sake of comparison, the two domains modeled with FRAGs can generate the same solutions as the base domain. The basic domain, which we will call no-frag, has 4 actions which are $load$, fly, $unload$ and $refuel$. The second domain called lo–fly–$unlo$ contains a FRAG with $load$, fly and $unload$ operator nodes and can infer all possible sequences of these 3 actions (depending on FRAG initialization). Finally, the third domain called $refuel$–fly contains the $refuel$ and fly operator nodes.

Figure 4 presents the results we obtained on the top 15 problems used in international planning competitions. The bottom graph shows that the use of FRAGs does not affect significantly the total cost of the solutions. To focus only on the comparison with and without FRAGs, the cost of each action is 1, which makes the total cost equal to the size of the plan because these are 3 sequential problems. The top graph gives the time to execution (in seconds) and the middle graph gives the number of nodes explored during the search (both scales are logarithmic). Let's start by comparing the $refuel$–fly domain and the no-frag domain. The number of

[1]http://ipc02.icaps-conference.org/

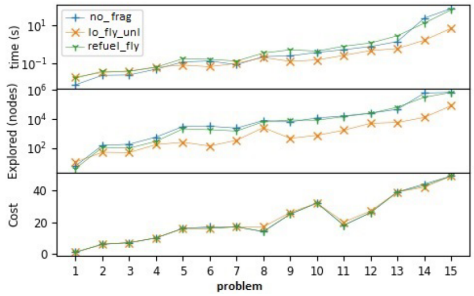

Fig. 4. Comparison test with and without the use of FRAGs

nodes explored is about the same and the execution time is slightly higher for the refuel-fly domain. We therefore have an additional processing time FRAGs but which does not seem significant. This is explained by the fact that the FCMs execute quite quickly and especially that the result of the processing of the FRAGs is stored in a dictionary, which greatly reduces the number of calls to the FRAG processing function. We now compare the $lo–fly–unlo$ domain and the no-frag domain. We can see that unlike the refuel-fly domain, the $lo–fly–unlo$ domain crawls fewer nodes and runs faster than the no-frag domain. This is explained by the fact that the actions load then fly or fly then unload often appear consecutively in solutions. This effect does not appear on the refuel-fly domain because the refuel action is rarely used and therefore the $refuel \leftarrow fly$ sequence will rarely be needed. The sequences of actions generated actually behave like the macro-actions proposed by [22], which are macro-operators that group at least two actions to insert them all at once into the plan and thus accelerate the search. To sum up, the processing of FRAGs effectively generates additional time, which remains however low. Moreover, this additional time can be compensated by a reduction in the number of explored states if the sequences of actions generated by the FRAGs often appear in the solutions.

B. Impact of FRAGs on planning

The particularity of our system is to plan in the space of actions and FRAGs whereas planning is usually done only in the space of actions. This is why we sought to estimate the impact in execution time of FRAG processing on the total planning time. To do this, we calculated the percentage of time spent in the FRAG Operator Processing compared to the total planning time. We performed these tests on the DriverLog domain[2] which consists of delivering packages to different locations with trucks. The actions are unloading and loading the truck, getting the driver on and off the truck, moving the truck from point A to point B, etc. We have grouped three of these actions into a FRAG: *load*, *drive* and *unload*. This FRAG can generate the following action sequences: $\{load \rightarrow drive \rightarrow unload\}$, $\{load \rightarrow drive\}$, $\{drive \rightarrow unload\}$, $\{load\}$, $\{drive\}$, $\{unload\}$.

[2]http://ipc02.icaps-conference.org/

TABLE I
ESTIMATION OF FRAG PROCESSING TIME ON TOTAL PLANNING TIME

Pb	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Time (%)	18	19.5	17.7	12.4	14.7	8.3	10.4	13.7	12.3	15.7	7.5	2.8	4.4	2.4	1.8

Table I shows the results we got on the top 15 problems in the DriverLog domain (listed in order of complexity). The second row of the table shows the percentage of time spent in the processing function of a FRAG compared to the total planning time. We can see that this percentage is of the order of 15-20To understand this result, the sequences of actions generated by the processing function of a FRAG are stored in a dictionary for a given initialization, and this avoids processing a FRAG several times for the same initialization. The consequence of this is that after a while, no more calls to the processing function of a FRAG are necessary and the planner finds himself planning in the space of actions and the space of sequences of actions generated.

VIII. CONCLUSION

In this paper, we presented an orchestration system based virtual environment to train a medical leader to manage a mass casualties influx. To learn to manage these situations, we assumed the trainee must be exposed to a wide range of situations and of team. These situations must be controlled. Then, we leverage these different contradictory objectives thanks to **a compromise between autonomy and scripted approaches by coupling different approaches 1) a numerical approach with a fuzzy graphical models (FCM), and 2) a symbolic approach with planning techniques**. Our approach includes a balancing between these objectives. We proposed a FCM to script learning situations which are instantiated on the fly to create more adaptability and to allow control and user agency. We proposed a dynamic planner to contextualize the situations and to allow the resilience of the system and the adaptability. The authors need to write specific fragments and it is still a hard work. Then in the future, it could be interesting to generate automatically new fragments according to pedagogical constraints and to semantic proximities. The time is not considered here and it is intended, the scenario unfold depending on it's pre-conditions, similarly to a PDDL plan. The actions of the characters are taken into account as their actions will unlock pre-conditions, which in turn will unlock events of the plan. The scenario is expected, since the beginning, actions from the characters and eventually some branches of the scenario will not occur if the characters does not behave as expected, in which case an online replanning mechanism can be triggered. In the future, it might also be interesting to manage the temporal constraints in the orchestration engine.

ACKNOWLEDGMENT

The authors would like to thanks the ANR and the DGA for the funding of these works.

REFERENCES

[1] M. Jacob and B. Magerko, "Interaction-based authoring for scalable co-creative agents." in *ICCC*, 2015, pp. 236–243.

[2] B. Magerko, R. E. Wray, L. S. Holt, and B. Stensrud, "Improving interactive training through individualized content and increased engagement." in *Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2005.

[3] M. Si, "Thespian: a decision-theoretic framework for interactive narratives," Ph.D. dissertation, University of Southern California, 2010.

[4] C. Martens, O. Iqbal, S. Azad, M. Ingling, A. Mosolf, E. McCamey, and J. Timmer, "Villanelle: Towards authorable autonomous characters in interactive narrative." in *INTWICED@ AIIDE*, 2018.

[5] S. G. Ware *et al.*, "A plan-based model of conflict for narrative reasoning and generation," 2014.

[6] M. Cavazza, F. Charles, and S. J. Mead, "Character-based interactive storytelling," *IEEE Intelligent Systems*, vol. 17, no. 4, pp. 17–24, 2002.

[7] D. Pizzi and M. Cavazza, "Affective storytelling based on characters' feelings," in *Proceedings of the AAAI Fall Symposium on Intelligent Narrative Technologies*, 2007.

[8] J. Niehaus and M. Riedl, "Scenario adaptation: An approach to customizing computer-based training games and simulations," in *AIED*, 2009, p. 89.

[9] M. O. Riedl, A. Stern, D. Dini, and J. Alderman, "Dynamic experience management in virtual worlds for entertainment, education, and training," *International Transactions on Systems Science and Applications*, vol. 4, no. 2, p. 23–42, 2008.

[10] R. Aylett, R. Figueiredo, S. Louchart, J. Dias, and A. Paiva, "Making it up as you go along-improvising stories for pedagogical purposes," in *Intelligent Virtual Agents*, 2006, pp. 304–315.

[11] R. Hill, J. Gratch, W. L. Johnson, C. Kyriakakis, C. LaBore, and R. Lindheim, "Toward the holodeck: integrating graphics, sound, character and story," in *Proceedings of the fifth international conference on Autonomous agents*, ser. AGENTS '01. New York, NY, USA: ACM, 2001, pp. 409–416.

[12] H.-M. Chang and V.-W. Soo, "Planning to influence other characters in agent-based narratives," in *Integrating Technologies for Interactive Stories Workshop, International Conference on Intelligent Technologies for Interactive Entertainment*, 2008, pp. 12—17.

[13] M. O. Riedl and V. Bulitko, "Interactive narrative: An intelligent systems approach," *Ai Magazine*, vol. 34, no. 1, pp. 67–67, 2013.

[14] P. Wang, J. P. Rowe, W. Min, B. W. Mott, and J. C. Lester, "Interactive narrative personalization with deep reinforcement learning." in *IJCAI*, 2017, pp. 3852–3858.

[15] S. Teffali, N. Matta, and E. Chatelet, "Managing stress with experience feedback in crisis situation," *AI EDAM*, vol. 33, no. 2.

[16] B. Kosko, "Fuzzy cognitive maps," *International Journal of Man-Machine Studies*, vol. 24, no. 1, pp. 65–75, 1986.

[17] I. Miguel, P. Jarvis, and Q. Shen, "Flexible graphplan," in *ECAI*, 2000, pp. 506–510.

[18] J. Porteous, "Planning technologies for interactive storytelling," in *Handbook of Digital Games and Entertainment Technologies*. Springer, 2016.

[19] M. Cavazza and R. M. Young, "Introduction to Interactive Storytelling," in *Handbook of Digital Games and Entertainment Technologies*, 2016, ch. 17, pp. 377–392.

[20] A. Shoulson, M. Kapadia, and N. I. Badler, "Paste: A platform for adaptive storytelling with events," in *Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2013.

[21] J. O. Ryan, M. Mateas, and N. Wardrip-Fruin, "Open design challenges for interactive emergent narrative," *Lecture Notes in Computer Science*, vol. 9445, pp. 14–26, 2015.

[22] A. Botea, M. Enzenberger, M. Müller, and J. Schaeffer, "Macro-FF: Improving AI planning with automatically learned macro-operators," *Journal of Artificial Intelligence Research*, vol. 24, pp. 581–621, 2005.