# Validating the plot of Interactive Narrative games

Carolina Veloso
*INESC-ID*
*Instituto Superior Técnico,*
*Universidade de Lisboa*
Lisboa, Portugal
carolina.veloso@tecnico.ulisboa.pt

Rui Prada
*INESC-ID*
*Instituto Superior Técnico,*
*Universidade de Lisboa*
Lisboa, Portugal
rui.prada@tecnico.ulisboa.pt

***Abstract*—The authoring of interactive dialogues in video games is an overwhelming and complex task for game writers. Developing an Interactive Narrative that balances authorial intent and players' agency requires frequent in-depth testing. The limited range of tools to assist authors in verifying their story can limit the creation of more complex narratives. In this paper, we discuss the challenges of Interactive Story design and provide a model consisting of a set of metrics for testing interactive dialogues. Using this model, we developed a prototype for the *Story Validator* tool. This debugging tool allows game writers to experiment with different hypotheses and narrative properties in order to identify inconsistencies in the authored narrative and predict the output of different playthroughs with visual representation support. We conducted a series of user tests, Using the *Story Validator*, to investigate whether the tool adequately helps users identify problems that appear in the game's story. The results showed that the tool enables content creators to easily test their stories, setting our model as a good step towards automated verification for assistance of authoring interactive narratives.**

***Index Terms*—Interactive Narratives, Authoring Systems, Verification, Automated Playtesting**

## I. Introduction

The presence of an immersive story in video games is often a central part of game experience and, consequently, a concern of design [1] [2]. Such games combine interactivity and storytelling as an Interactive Narrative (IN), often nonlinear, that allows the player's actions to alter the course of the story [3] [4]. At the core, IN functions similarly to the Choose-Your-Own-Adventure storybooks [5], where the reader is faced with various decision points at which they must make a choice, branching the story in different directions, often leading to different outcomes.

Deploying IN in games has enormous potential for enabling the creation of interactive systems that combine player interaction and dynamic plots, however, authors face many challenges when writing this type of stories [6]. Developing an IN where players can feel immersed and engaged involves making the player's actions and choices have a powerful influence on the narrative's direction, making it challenging for the author to guarantee a well-formed story. Each branch in the story has to be carefully tailored by the game's writer, who consequentially has to predict the different possible player's behaviours that can affect the story at various states so that they can present those choices to the player.

Most traditional approaches rely on extensive and rigorous playtesting to obtain information on how the players experience the narrative and what might need improvement. Playtesting provides insightful information that is not anticipated during development, helping authors ensure that both the game's narrative and the player's behaviour are well-balanced [7] [8]. However, obtaining quality feedback for a detailed analysis can be challenging, expensive, and time-consuming [9]. Various works have offered different solutions to handle narrative conflicts caused by user behaviour in-game. However, hardly any works have focused on letting the author simulate and question their narrative during development. Instead, they opt for online AI approaches (such as drama managers [10]) that, during gameplay, provide ways to dynamically adapt the narrative and resolve conflicts created by unintended player's actions. These approaches create changes to the narrative that the author might not have intended, leading them to lose control over their story.

On the other hand, most authoring tools, such as Twine [11] and Ren'Py [12], while they facilitate the creation of IN, they lack the proper tools to identify possible continuity errors, to keep track of specific narrative properties and to envision the output of playthroughs prior to human playtesting. Instead, these tools rely heavily on the author's intuition to foresee these challenges or on an exhaustive number of later playtests.

This work outlines the development of a tool that supports game writers in creating IN whilst maintaining the human author's directorial control. Essentially, we strive to maintain the idea of authorial intent [13] and develop a system that allows human authors to express their artistic intentions without feeling constrained. As a product of this work, we have designed, developed, and tested a prototype for the *Story Validator*, which traverses all possible narrative paths of a IN and provides insightful data on different story metrics and design issues encountered via visual representations.

This work was supported by the iv4XR (Intelligent Verification/Validation for Extended Reality Based Systems) project, EU H2020-ICT-2018-2020/856716, and by the National Funds Through FCT, Fundação para a Ciência e a Tecnologia, under project UIDB/50021/2020.

## II. Authoring Interactive Narratives

The authoring of interactive stories in video games is an overwhelming and complex task for game writers, both in narrative design and implementation. We identified four main challenges pertaining to the development of IN:

1) **Authorial intent vs player agency.** One of the most challenging problems with IN is the necessity to balance authorial intent and player agency in the context of storytelling [14]. According to Riedl, authorial intent is the trajectory that the game writer wishes the player to follow, regardless of how the player acts during the game. Because IN allows the user to interact freely within the story world, users often have the power to behave in ways that are inconsistent with the plot. This can either prevent the plot from advancing or make the player experience the story in ways that the creator did not intend.
2) **Narrative exponential growth.** As the plot grows in complexity and the number of decision points increases, the authoring experience will often require substantial changes to keep the narrative coherent, which can become overwhelming for the author [15]. Not to mention that the impact a choice has may end up only revealing itself in future states of the story, which is not always easy to predict.
3) **Gameplay variables that affect the story.** Besides what choices the player makes, the story's development can be based on different variables and status that are updated throughout to game. A common example is the "karma systems", which consider the player's good and evil deeds and shape the world around them as they progress. In more complex narratives, game variables can become challenging to keep track of, and their consequences may only unravel in later states in the game, which is not always easy for the author to foresee during development.
4) **Measuring the impact on user's experience.** The player play-styles are reflected not only in how they interact with the narrative but also in how they expect the story to unfold [16]. Due to the complex nature of IN, it is difficult for the author to predict the player's behaviour and overall emotional experience. This is challenging to predict without prior human playtesting, as it is hard to ensure during development.

## III. Literature Review

### A. Interactive Drama

In order to maintain narrative coherency even when the player has freedom of choice, some past approaches have focused on the idea of integrating real-time Artificial Intelligence (AI) methods to shape the narrative, allowing players to interact with the game freely but, at the same time, making sure the narrative still follows a coherent structure. A popular example, first proposed by Bates [17], is the use of a drama manager. A Drama Manager (DM) is an omniscient agent that monitors the game world and guides the player's experience through the story by searching for possible future plot points based on the evaluation of the current game world while still allowing the players to interact freely.

For instance, one of the most famous examples of the implementation of a DM is the Mateas and Stern interactive drama Façade [18]. Façade uses its DM to monitor and update the simulation in real-time in response to text the user inputs by selecting the next story beat.

Like in Façade, Riedl et al. presented the prototype for INTALE [19], in which the agents are directed by a DM called Automated Story Director. The Automated Story Director handles user interactions by maintaining a script of expected events and planning out new narrative follow-ups to respond to the player's actions and achieve all concrete narrative goals pre-defined by the author.

The studies mentioned above show that online AI approaches, particularly drama managers, are a popular solution to guarantee narrative coherence while allowing the player to interact with the game freely. Unfortunately, there has been hardly any work done regarding off-line approaches that attempt to identify possible narrative problems during development and allow the author to validate their narrative with different story metrics and predict different playthroughs' output.

### B. Authoring Tools

Since the authoring process remains one of the most significant challenges in the development of interactive stories, there is a need for tools, toolkits and authoring systems that assist game writers in creating their content. Authoring tools for IN provide different visual representations and structural elements that allow authors to write their creative works.

With this in mind, we explored some of the existing authoring tools currently available for creating IN and their limitations, namely Tinderbox [20], Ren'Py [12], FAtiMA Toolkit [21] and Twine [11]. Our findings revealed that these authoring systems lacked the proper tools to analyse important narrative metrics and identify possible continuity errors. While these tools were designed to facilitate the authoring process, they nevertheless rely heavily on the author's diligent and methodical eye to predict all gameplay scenarios and possible complications that might arise, making story creation a wearing, expensive, and time-consuming process.

Nonetheless, due to its wide adoption and easy to work architecture, we concluded that Twine was the most promising candidate for the incorporation of our model, which we will describe in the next section.

**Twine** is a hypertext-based authoring tool designed to facilitate the creation of interactive stories with branching narratives. The creation of games with Twine requires only two elements: *Passages* and *Links*. *Passage* is the Twine's terminology for the nodes on the story graph that players navigate through. Each *Passage* contains a block of text that is shown to the player when they reach that *Passage* during gameplay. Additionally, *Passages* can also possess one or more

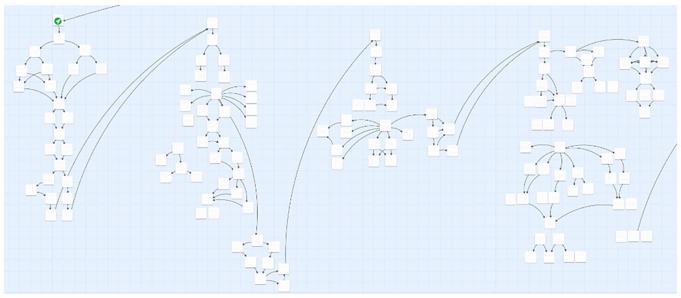


Fig. 1. A bird's-eye view of a storyboard in Twine.

tags, which function as labels that add information to the *Passage* but are not visible on the published version of the story.

Similar to branching narratives, where arcs connect nodes to each other, *Passages* are connected through *Links*, represented by an arrow on Twine's storyboard. To the player, these are displayed as points of interaction or decision points. The *Links* can be guarded by variables. Twine has a <<*if*>> macro, which is used to test the current value of the story's variables. These macros will influence the flow of the story since certain *Links* will only be available to the player if the conditions of the <<*if*>> macros are true.

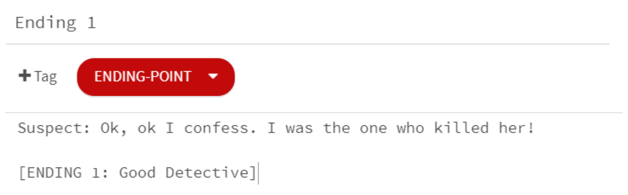
Because Twine is designed to develop and publish interactive stories on the Web, all its data is encoded in a single HTML file. However, while HTML is the default hypertext output for Twine, we found that exporting Twine's internal XML data in a JSON format would be an added value for reasons of simplicity and easier processing. This JSON file contains each *Passage*'s data, including the *Passages*' text, name, id, *Links*, and position on Twine's storyboard.

## IV. A Model to Test Interactive Narratives

We sought out to develop a tool that supports game writers in the creation of IN in stages before human playtesting by letting the author explicitly test different hypotheses and narrative properties to identify possible design mistakes. To be useful to authors, such a tool needs to provide insightful data on different story metrics and identify design issues that a player could encounter during gameplay.

We take the assumption that the story space can be represented by a traversable graph with nodes (*Passages*) and edges (*Links*) that can be guarded by variables, and that the tool uses a Depth-First Search (DFS) based algorithm to traverse all possible narrative paths to gather data. With this in mind, we defined the following as useful narrative metrics:

- **Number of paths.** Enumeration of all possible traversals of the story graph, including the *Passages* the player visits on each narrative path. As a result with get the number of visits to each *Passage* as well and get an idea about which parts of the story are probably more often experienced by the players. When searching the graph, the algorithm begins at the starting node (obtained from the JSON data) and explores one *Link* at a time, adding each *Passage* visited to a stack. When it reaches an ending, the total number of paths is increased by one, the elements in the stack are saved as a path sequence, and we begin popping nodes from the stack until we find a node with an unvisited *Link*.
- **Endings Hit Percentage.** At the end of the game, the story reaches different endings, depending on the players' choices and state variables. The distribution of the endings players reach is often a concern of the design. As the search algorithm traverses all the paths, it keeps count of how many paths reach each ending. It then calculates the corresponding hit percentage. To ensure that, the story ending nodes need to to be identified. For example, by using the Twine tag system and adding an ENDING-POINT tag to the *Passages* that are defined as an ending (see Fig. 2).



Ending 1

Suspect: Ok, ok I confess. I was the one who killed her!

[ENDING 1: Good Detective]

Fig. 2. Example of an ENDING-POINT tag in Twine.

- **Stroke Points.** Whenever the search algorithm reaches an ending, we compare the current path sequence with the previous one, intersecting their values to find common nodes in all paths. Eventually, we can identify which *Passages* are visited in all narrative paths. We named these *Passages*, Stroke Points. These can be part of the designer goals, as they can be useful to ensure some parts of the story is always conveyed to the player. However, it may happen that the author does not want to withdraw the player's ability to choose, in which case Stroke Points may become a problem.
- **Lost Plot.** Identifies narrative sections that, although included in the story by the author, are never reached in an actual playthrough due to some design or implementation error, for example, the conditions set in the <<*if*>> macros are never met. Ideally, in a well-constructed interactive story, all *Passages* should be visited at least once, and all paths should reach an ending. As the algorithm traverses the story, it keeps track of the *Passages* visited. Eventually, it can identify the existence of *Passages* that are never reached by any possible narrative path. Additionally, the iteration of the search may end in Dead-End, as it stops once it reaches a node without any edges (i.e., a *Passage* without any possible *Links*). Normally, this node is expected to be an ending, but it can also be a Dead-End — a non-ending *Passage* that impedes the player from progressing. Therefore, if the end *Passage* does not have an ENDING-POINT tag, it is identified a Dead-End.
- **Evolution of Variables.** A branching narrative might

contain different variables that the author needs to monitor. As the algorithm visits each *Passage*, it keeps track and updates those variables. This is used in the tool to display the values of the variables in a given path and to trace a timeline of the variable variations through a given playthrough.

The above-mentioned metrics were defined based on our personal experience as developers of interactive stories using Twine and based on a reflection to address the authoring problems discussed in section II. The same challenges were later reported by the user study's participants in section VI-D, which corroborated our ideas.

## V. Story Validator prototype

The implementation of the *Story Validator* tool follows the architecture presented in Figure 3. It is able to load an *interactive story* (in JavaScript Object Notation (JSON) format) created using Twine's authoring environment. It creates a representation of the story as a branching tree that is traversed by a Depth-First Search (DPS) based algorithm. It uses a DFS algorithm to be greedy in the exploration to explore a given story path until the end, as a human player would experience the story (i.e., starting at the root *Passage* and traversing through several *Passages* until reaching an ending point). The information collected is presented to the user through the use of a graphical interface (GUI), depicted in in Figure 4.

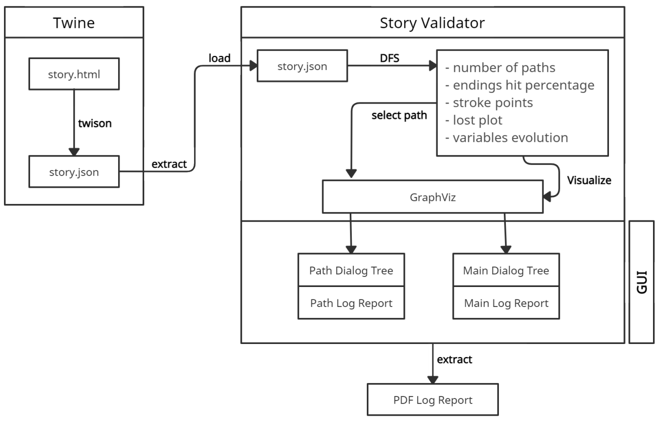


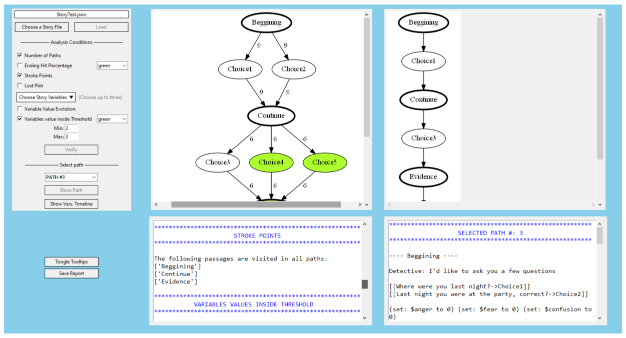
Fig. 3. The general architecture of the Story Validator tool.

The prototype was built in Python 3 using tkinter for the Graphical User Interface (GUI)[1]. The tool's GUI was developed following the conceptual organisation presented in Figure 5[2].

Using the *Story Validator's* GUI, the user selects (1) a JSON story file to be analysed by the tool. For this analysis, the user picks one or more options from a selection of analysis conditions (2), based on the metrics defined in section IV

[1]The source code can be found at https://github.com/iv4xr-project/in-story-validator

[2]Note that elements 4, 9 and 10 were added only after Phase 1 of user testing.


StoryTest.json

Analysis Conditions

[✓] Number of Paths
[ ] Ending Hit Percentage: green
[✓] Stroke Points
[ ] Lost Plot
Choose Story Variables (Choose up to three)
[ ] Variable Value Evolution
[✓] Variables value inside Threshold: green
Min: 2
Max: 3

Select path: PATH #3

```
********************************************************
                    STROKE POINTS
********************************************************

The following passages are visited in all paths:
['Beggining']
['Continue']
['Evidence']

********************************************************
          VARIABLES VALUES INSIDE THRESHOLD
********************************************************
```

```
********************************************************
                 SELECTED PATH #: 3
********************************************************

---- Beggining ----

Detective: I'd like to ask you a few questions

[[Where were you last night?->Choice1]]
[[Last night you were at the party, correct?->Choice2]]

(set: $anger to 0) (set: $fear to 0) (set: $confusion to
0)
```

Fig. 4. The Story Validator prototype GUI.

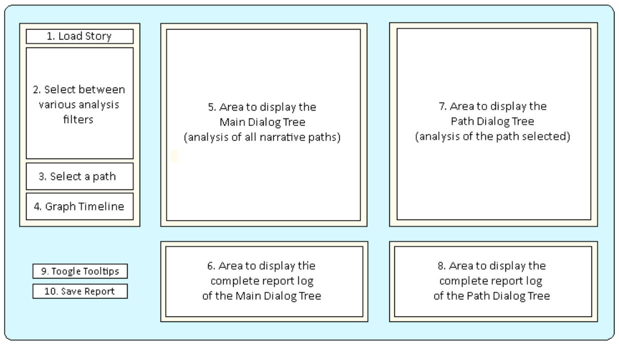
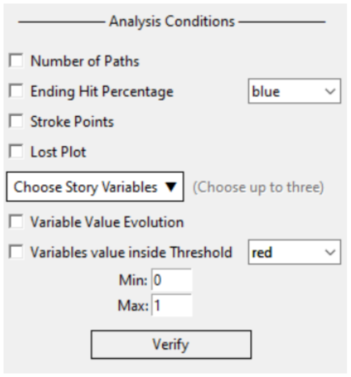

1. Load Story
2. Select between various analysis filters
3. Select a path
4. Graph Timeline
5. Area to display the Main Dialog Tree (analysis of all narrative paths)
7. Area to display the Path Dialog Tree (analysis of the path selected)
9. Toogle Tooltips
10. Save Report
6. Area to display the complete report log of the Main Dialog Tree
8. Area to display the complete report log of the Path Dialog Tree

Fig. 5. The Story Validator GUI conceptual structure.

(see Fig. 6). Each one of these conditions will influence different visual aspects of the Dialog Trees (5 and 7) and what information is shown on the Log Reports (6 and 8).


Analysis Conditions

[ ] Number of Paths
[ ] Ending Hit Percentage: blue
[ ] Stroke Points
[ ] Lost Plot
Choose Story Variables (Choose up to three)
[ ] Variable Value Evolution
[ ] Variables value inside Threshold: red
Min: 0
Max: 1

Fig. 6. The panel of analysis conditions.

Additionally, the user can select a story path to analyse in detail which will appear on the Path Dialog Tree (7) area.

As stated previously, this work's main objective is to support game writers by evaluating their game's narrative and working as a debugging tool. Next we will present how we envision authors using our prototype tool to identify and solve problems present their IN.

- **Keeping track of Passages visited.** Often during the creation of IN, there is the need to add, change and remove *Passages* and this process can quickly impact the

dynamic of the story. However, this is not always easy to predict as the impact of a *Passage* can reveal itself only in future states.
Furthermore, a branching narrative will often have several narrative paths that the player can take, resulting from their choices. For that reason, as the number of possible narrative paths grows, it becomes tougher for the author to keep track of the *Passages* that are visited.
The tool's algorithm runs through the story by selecting different options until it reaches an ending. Then it starts a new playthrough and repeats, choosing other options. In the end, the Main Dialog Tree (5) displays a tree with all the narrative paths that the player can take, giving the author a general idea of how the story flows. In order to make a more in-depth assessment, the Main Report Log (6) displays which *Passages* are visited in each path. Additionally, the user can pick a path to analyse in detail (3), which is displayed on the Path Dialog Tree (7).

- **Keep track of endings' reachability.** Depending on the choices in dialogue made, the player is led to different endings. However, it is difficult for the author to predict and monitor the endings' distribution without it being a laborious and time-consuming task. As a design objective, the author may want to create certain restrictions on the distribution of the endings, such as having an ending that is more common to obtain or an ending with only one possible path. If the analysis condition *Endings Hit Percentage* is selected, the Main Report Log (6) then provides the author with percentages on the likelihood of reaching each ending. Besides, on the Main Dialog Tree (5), the user can observe the paths' distribution if the analysis condition *Number of Paths* is selected.
- **Keep track of the story's variables.** In IN, the path in which the narrative develops can be dependent on different game's variables. Having variables updating depending on the characters or plot state can make the interactive story more interesting for the player; however, it is difficult for the author to keep track and control those values over multiple interactions. The tool can identify and keep track of all variables defined by the author and their values throughout each path. The user can also select which variables they wish to analyse (up to three). By selecting the analysis condition *Variables Value Evolution*, the user can observe the value changes of each variable. By selecting the analysis condition *Variables value inside Threshold*, the tool highlights the *Passages* where the variables have a value between the MIN and MAX values, both defined by the user. The story variables are also presented in a dotted chart (4), displaying the value's changes throughout the story.
- **Avoiding Dead-Ends and losing plot.** Dead-Ends typically happen due to an error in defining the *Passages* that should come afterwards, meaning they have "entering" conditions that are impossible to meet or if the path is an "endless" path. The designer must identify these cases, as they abruptly stop the player from continuing playing. However, doing so is difficult due to the combinatorial nature of the exploration of the story. Furthermore, it is also essential to identify if there are sections in the story that are never visited, at the risk of losing important parts of the plot. If the user selects the analysis condition *Lost Plot*, the Main Report Log (6) will print out the *Passages* that were never visited and display which paths were not able to reach an ending. Additionally, on the Main Dialog Tree (5), *Passages* that are never visited will appear as single nodes with no edges connected to them, and on the Dialog Trees (5 and 7), Dead-End nodes will take on a rectangular shape in oppose to its regular round shape, so they are easier to identify for the user.

## A. *Emulating personas and playstyles*

As mentioned in section II, the players' play-styles are reflected not only in how they act within the plot, but also on how they expect the story to respond to their actions.

The author may want to guarantee that some types of players follows a particular trajectory. Therefore, our intent was to develop an agent that simulates various playthroughs of the game, emulating the behaviour of different players, in order to predict user experience and study the story's output.

In their work, Stahlke et al. [23] developed a framework, named PathOS, that simulates human testing sessions using intelligent agents. These agents mimic the behaviour of human players by following a set of heuristics (e.g., curiosity, aggression, and completion), that are configured to reflect different play-styles. Our work proposes a similar approach, by letting the author assign different weights to each of the story variables to create different personas that represent different playstyles. These personas will have a tendency to favour options in the story that have stronger impact in certain variables.

For this, we use an informed **Greedy search** (see Algorithm 1) using as heuristic based on the defined weights for the "persona" under test. Each story has a set of variables $V = (v_1, v_2, ..., v_n)$, where *n* is the total of story variables, and to each of these variables *V*, the writer assigns weights $W = (w_1, w_2, ..., w_n)$. The heuristic then drives the playthrough of the story to give preferences to *Links* that raise the value of variables it gives higher weight.

For example, if we treat the variables as "emotional states", the author may study how the story unfolds given the behavior of a more aggressive type of player in their game scenario. They can give greater weight to an "angry" variable that depicts the emotion of a character in the story and, consequently, drive the algorithm to choose the options that lead to higher value to that emotion. By specifying different weights to each story variable (e.g., 75% fear, 15% confusion, 5% angry), the author creates agents with distinct preferences that play the game in different ways.

It is important to note that since the greedy algorithm does not consider the overall problem and always makes a locally-optimal choice at each step based on the information it has, it often does not produce the most optimal solution. However,

**Algorithm 1:** Greedy search using heuristic based on the defined weights.

```
Function Greedy_step(hightest_total, passage, V, W):
    /* gather all the passage's links */
    $links = passage.get("links")$
    for $l$ in $links$ do
        /* see that $v_{nl}$ is the value update */
        /* of the variable $n$ on link $l$ */
        $total_l = v_{1l} * w_1 + v_{2l} * w_2 + \ldots + v_{nl} * w_n$
    end
    /* chose link that maximizes the total value */
    /* assume $x$ is the total number of links */
    $chosen\_link = \mathbf{max}(total_{la}, total_{lb}, \ldots, total_{lx})$
    /* update highest total value */
    $highest\_total = highest\_total + chosen\_link$
    /* chose link with highest value to visit */
    $next\_passage = l_{max}$
    /* move to next iteration */
    $Greedy\_step(highest\_total, next\_passage, V, W)$
```

we need to remember that, in most cases, the human player is also unaware of the overall game-space during gameplay and, therefore, makes the choices they believe to be the most optimal at each step, based on the information they currently have and their player profile. Thus, we believe that a greedy algorithm best emulates the human behavior in this scenario.

## VI. Experimental Results

The following section presents the methods and results of two user studies that were conducted with the intent of: finding out if the tool adequately helps the users identify problems in the game story, and determining whether users can operate the tool with ease and identify usability issues.

### A. The Scenario

For the tests we used Twine's authoring tool to create an *interactive story*, which we named *"The Murder Case"*.

The story has the following setting: the player plays a detective tasked with solving a murder. The detective's main suspect is brought to the police station for questioning and, since existing evidence is not sufficient, the player will need a confession. However, there is a catch: the suspect NPC is a very wealthy man and upsetting him during questioning will lead him to ask for his lawyer and the detective to lose the case. Depending on the dialogue choices made when talking with the suspect, the player is led to different endings that reflect their crime-solving skills. To stipulate the effect the player's choices have in the narrative's unfolding, we attach (and update) variables to each dialogue option. These variables represent the different emotional states of the suspect NPC: *anger*, *confusion*, and *fear*.

The scenario contains 3 different endings ("Good detective", "Lawyered" and "Not enough evidence"), 14 *Passages* and 20 possible narrative paths players can take.

Additionally, we created an alternative version of the story with added issues to be used in the User Test 2. This version, named *"TMC - failed version"*, includes 2 *Passages* that are never visited and 3 Dead-Ends.

### B. User Study 1

In the first study, we examined how the users feel about the tool's design and checked its usability. The study was conducted with 5 users (3 female and 2 male), ages between 18 and 24. Most participants admitted they played games either regularly (40%) or every day (40%), with some (60%) enjoying games with interactive stories and branching narratives. Out of the 5 participants, 3 stated they had knowledge in game development, manly working as a Game Programmer and/or Game Writer.

The participants were asked to perform 10 tasks using the *Story Validator* and *"The Murder Case"* scenario. These tasks included calculating the total number of paths, finding the endings' distribution, studying the story variables' evolution in each path, among others. This version only allowed users to test one story variable at a time.

While the participants completed the tasks, we observed their performance and took notes. We used the think-aloud method, where participants used the system while continuously verbalising their thoughts.

In general, the participants were able to complete the tasks, but some difficulties were found in the tasks that required checking the values of variables. For example, two participants did not complete a task that involved counting the number of paths that reached an ending with a variable "anger" = 0, while one was not able to find *Passages* in which the variable "anger" had values between -4 and -2.

The participants found the *Story Validator* tool straightforward and easy to use, giving it an average score of 83.5 (SD = 9.45) in the System Usability Scale (SUS). According to a study made by Bangor, Kortum, and Miller [22], these results suggest that tool has "passable" overall usability.

In the study, participants also proposed different suggestions regarding the tools' usability, for example:

1) **Help understanding how the tool works.** Some users noted that the quantity of features was a bit intimidating ("[the tool] is very overwhelming at first"). It was suggested some form of assistance to explain the tool's functionality ("Once you start clicking some checkboxes, you learn pretty quickly how it works [...] but having a help button or something similar would have helped a lot.").
2) **Easier to read log report.** In cases where multiple analysis conditions were selected, users had issues while navigating through the log report box and often had to keep scrolling up and down to locate what they were seeking. Participants noted that if the box was larger it would be easier to navigate.

3) **Ability to analyse more than one variable.** One of the users reported the desire to analyse more than one variable at a time. While this did not hinder their ability to complete the tasks, their suggestion was noted down.

Based on the feedback gathered, we improved the prototype and added some new features, including:

- The option to test more than one variable simultaneously.
- A "Toggle Tooltips" button, which allows the user to hover different elements on the tool's menu to display a short message detailing how each works.
- A "Print Report" button, which creates a full pdf report of the story analysis that the user can access outside the tool's workspace.
- A dotted chart graph, which displays a timeline with the changes in the values of each story variable select for analysis throughout the story path selected.

## C. User Study 2

For the second user study we used an improved version of the *Story Validator* that addressed the usability issues found in the first study and included a few extra features.

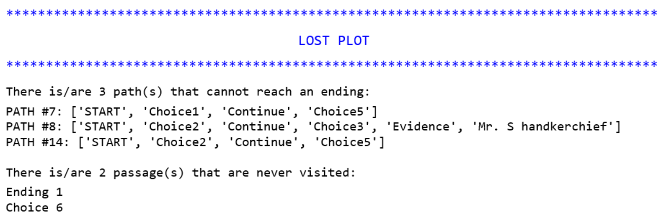
The goal of this study was to test if users were able to use the *Story Validator* to identify problems and various design issues in the interactive story (the *"TMC - failed version"*), and suggest possible solutions to the problems they find.

We conducted the test with 20 participants (8 female and 12 male). Except for one participant, all others admitted they played video games either every day (35%) or regularly (60%). Nine of the twenty participants said to have some knowledge in game development (as Game Programmers, Game Designers or Game Writers). Five of those participants were familiar with Twine's authoring tool prior to the test and considered themselves to have "Intermediate" expertise.

The scenario that participants tested had the following issues:

- **Problem 1:** The Path #7 does not reach an Ending.
- **Problem 2:** The Path #8 does not reach an Ending.
- **Problem 3:** The Path #14 does not reach an Ending.
- **Problem 4:** The Passage "Ending 1" is never visited.
- **Problem 5:** The Passage "Choice 6" is never visited.

All the previous problems can be identified in the **Story Validator** under the Analysis Condition *"Lost Plot"*, which reveals *Passages* that can not be visited and Paths that do not reach an ending, if existent.


```
*********************************************************************************
                                   LOST PLOT
*********************************************************************************
There is/are 3 path(s) that cannot reach an ending:
PATH #7: ['START', 'Choice1', 'Continue', 'Choice5']
PATH #8: ['START', 'Choice2', 'Continue', 'Choice3', 'Evidence', 'Mr. S handkerchief']
PATH #14: ['START', 'Choice2', 'Continue', 'Choice5']

There is/are 2 passage(s) that are never visited:
Ending 1
Choice 6
```

Fig. 7. Analysis Condition "Lost Plot" for the "TMC - failed version".

However, the reasoning behind each of these problems is unique, and the user needs to make use of the tool in order to grasp what exactly is causing each of these issues.

For example, problems 1 and 3 have a similar cause: both reach a *Dead-End* at the *Passage* "Choice 5". Using the **Story Validator**, the user will realize that during the story's creation, the writer did not add any *Links* leaving *Passage* "Choice 5" and, consequently, the play-through stops abruptly there.

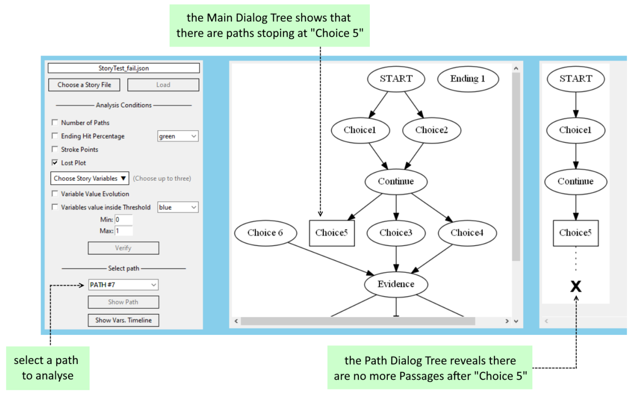

StoryTest_fail.json
Analysis Conditions
☐ Number of Paths
☐ Ending Hit Percentage: green
☐ Stroke Points
☑ Lost Plot
Choose Story Variables (Choose up to three)
☐ Variable Value Evolution
☐ Variables value inside Threshold: blue
Min: 0
Max: 1
Select path: PATH #7

Fig. 8. Analysing problems 1 and 3.

In general, the participants were able to identify the problems, except for three who could not identify problem 2. We believe that the reason for the misidentifying of this problem was that instead of using the Analysis Condition *"Lost Plot"* to identify the issues within the story, they found them by observing the tree displaying the in tool directly. While this method is legitimate, none of these 3 participants were able to identify and solve problem 2.

Additionally, two users (10%) could not find a solution for problems 1 and 3. In turn, problem 5 seemed to be the one that was harder to solve for at least five of the participants (25%). For all the other problems, the completion rate is 100%.

According to the average participant, the task of identifying the problems within the narrative was considered relatively easier than trying to propose a solution. The results also show the time it took users to identify all the problems was 52.3 sec. (SD = 22.66), while proposing a solution took, on average, 146.2 sec. (SD = 31.84). Overall, the time values for both finding and solving problems prove that the tool is efficient and that, with a bit of experience with the tool, users can quickly use the tool to pinpoint and repair errors in their interactive stories.

In the end, participants gave the *Story Validator* tool an average SUS score of 92.4 (SD = 4.76). These results suggest that our system is considered a "truly superior product" [22].

## D. Additional results

Before each experiment, we asked the participants, together with the demographics questionnaire, their preferred tool to write stories for games, and what problems they usually face in the task.

The most preferred authoring tool to write stories for games was a basic word processing tool (76%), such as Microsoft Word, Google Docs or LibreOffice Writer, but some had experience using Twine (24%). The users also reported the following problems when using their preferred authoring tool for writing IN:

- Difficulty keeping track of game variables and/or objects;
- Difficulty identifying "Dead-Ends", meaning moments in the game that prevent the player from continuing playing;
- Difficulty keeping track plot moments in the game that the player skips unintentionally, losing the plot coherence;
- The lack of ability to keep track of user experience (before playtesting).

These findings are aligned with the metrics we propose in section IV.

## VII. Conclusion

In this paper, we have underlined some of the current challenges concerning the authoring process of IN. We addressed the lack of tools that provided ways for the authors to test their narrative while considering the player's agency, during the game's developmental stage, without requiring playtesting. More often than not, these works opt for online AI approaches that, during gameplay, dynamically adapt the narrative and resolve conflicts created by unintended player's actions. This might lead to situations where the system takes control of the story, replacing human authorship.

With this work, we set ourselves to develop a tool for testing interactive dialogues for video games. With such tool, we believe that authors gain some control that enables them to define more complex narratives to express their artistic intentions without feeling constrained. This approach has been designed to facilitate the development of IN in stages before human playtesting by letting the author explicitly test different hypotheses and narrative properties to identify possible design mistakes. The tool's GUI allows for a clearer picture of the IN authoring process, by providing a visual representation of the narrative structure through the use of tree structures, that run through different test conditions. The two studies we conducted show some good sign for the approach and tool proposed. As several participants confirmed ("I would definitely use this tool to test my stories", "[the tool] really helps give the designer an idea of how their narrative works", "using [the tool] would save game developers so much time") the tool proves to be an essential asset for the creation of IN.

Overall, we believe that, as a first approach to this type of systems, our prototype managed to achieve the proposed objectives.

## Acknowledgments

We gratefully acknowledge the assistance and support provided by the colleagues of project iv4XR at INESC-ID. We would also like to thank all those who participated in the user studies; their feedback was extremely helpful.

## References

[1] E. Aarseth, "A narrative theory of games," in Proceedings of the international conference on the foundations of digital Games, 2012, pp. 129–133.

[2] H. Qin, P.-L. Patrick Rau, and G. Salvendy, "Measuring player immersion in the computer game narrative," Intl. Journal of Human– Computer Interaction, vol. 25, no. 2, pp. 107–133, 2009.

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

[4] S. Dinehart, "Dramatic play," 2009, online; Retrived 5-November-2020. [Online]. Available: http://www.gamasutra.com/view/feature/4061/dramaticplay.php

[5] B. Books, Choose Your Own Adventure Book Series. Bantam Books: NYC, 1979 - 1998.

[6] M. Mateas, "The authoring challenge in interactive storytelling," in Joint International Conference on Interactive Digital Storytelling. Springer, 2010

[7] P. Mirza-Babaei, N. Moosajee, and B. Drenikow, "Playtesting for indie studios," in Proceedings of the 20th International Academic Mindtrek Conference, 2016, pp. 366–374.

[8] P. Mirza-Babaei, V. Zammitto, J. Niesenhaus, M. Sangin, and L. Nacke, "Games user research: practice, methods, and applications," in CHI'13 Extended Abstracts on Human Factors in Computing Systems, 2013, pp. 3219–3222.

[9] N. Moosajee and P. Mirza-Babaei, "Games user research (gur) for indie studios," in Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems, 2016, pp. 3159– 3165.

[10] J. Bates, "Virtual reality, art, and entertainment," Presence: Teleoperators and Virtual Environments, vol. 1, no. 1, pp. 133–138, 1992.

[11] C. Klimas, "Twine - an open-source tool for telling interactive, nonlinear stories," 2020, online; Retrived 5-January-2020. [Online]. Available: https://twinery.org/

[12] T. Rothamel, "The ren'py visual novel engine," 2020, online; Retrieved 16-November-2020. [Online]. Available: https://www.renpy.org/

[13] M. O. Riedl, "Incorporating authorial intent into generative narrative systems." in AAAI Spring Symposium: Intelligent Narrative Technologies II, 2009, pp. 91–94.

[14] R. Aylett, "Emergent narrative, social immersion and "storification"," in Proceedings of the 1st International Workshop on Narrative and Interactive Learning Environments, 2000, pp. 35–44.

[15] E. Adams, Fundamentals of game design. Pearson Education, 2014, pp. 172–173.

[16] S. Dow, B. MacIntyre, and M. Mateas, "Styles of play in immersive and interactive story: case studies from a gallery installation of ar façade," in Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology, 2008, pp. 373–380

[17] J. Bates, "Virtual reality, art, and entertainment," Presence: Teleoperators and Virtual Environments, vol. 1, no. 1, pp. 133–138, 1992.

[18] Mateas, Michael, and Andrew Stern. "Façade: An experiment in building a fully-realized interactive drama." Game developers conference. Vol. 2. 2003.

[19] M. O. Riedl and A. Stern, "Believable agents and intelligent story adaptation for interactive storytelling," in International Conference on Technologies for Interactive Digital Storytelling and Entertainment. Springer, 2006, pp. 1–12.

[20] M. Bernstein, "Collage, composites, construction," in Proceedings of the fourteenth ACM conference on Hypertext and hypermedia, 2003, pp. 122–123.

[21] S. Mascarenhas, M. Guimarães, R. Prada, J. Dias, P. A. Santos, K. Star, B. Hirsh, E. Spice, and R. Kommeren, "A virtual agent toolkit for serious games developers," in 2018 IEEE Conference on Computational Intelligence and Games (CIG). IEEE, 2018, pp. 1–7.

[22] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," Intl. Journal of Human–Computer Interaction, vol. 24, no. 6, pp. 574–594, 2008.

[23] Stahlke, Samantha, Atiya Nova, and Pejman Mirza-Babaei. "Artificial Players in the Design Process: Developing an Automated Testing Tool for Game Level and World Design." Proceedings of the Annual Symposium on Computer-Human Interaction in Play. 2020.