Computational Narrative Blending Based on Planning

Edirlei Lima, Bruno Feijó, António L. Furtado

▶ **To cite this version:**

Edirlei Lima, Bruno Feijó, António L. Furtado. Computational Narrative Blending Based on Planning. 20th International Conference on Entertainment Computing (ICEC), Nov 2021, Coimbra, Portugal. pp.289-303, 10.1007/978-3-030-89394-1_22 . hal-04144376

HAL Id: hal-04144376

https://inria.hal.science/hal-04144376v1

Submitted on 28 Jun 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

This document is the original author manuscript of a paper submitted to an IFIP conference proceedings or other IFIP publication by Springer Nature. As such, there may be some differences in the official published version of the paper. Such differences, if any, are usually due to reformatting during preparation for publication or minor corrections made by the author(s) during final proofreading of the publication manuscript.

Computational Narrative Blending Based on Planning

Edirlei Soares de Lima[1,2][0000-0002-2617-3394], Bruno Feijó[3][0000-0003-4441-2632] and António L. Furtado[3][0000-0003-3710-624X]

[1] IADE, Universidade Europeia, Av. D. Carlos I 4, 1200-649, Lisbon, Portugal
[2] UNIDCOM/IADE, Av. D. Carlos I 4, 1200-649, Lisbon, Portugal
edirlei.lima@universidadeeuropeia.pt
[3] Department of Informatics, PUC-RIO, R. Marquês de São Vicente 225, Rio de Janeiro, Brazil
{bfeijo,furtado}@inf.puc-rio.br

Abstract. Inspired by conceptual blending models and considering plot generation as a plan-generation problem, this paper proposes a robust method that reuses existing stories to generate new narrative variants. This method generates variants that combine episodes extracted and adapted from different stories that share the same narrative structure. By combining a plan validation algorithm with a basic narrative structure, our method guarantees the logical coherence and general plot structure of the generated narratives. We also propose a new tool to assist amateur/professional writers to visualize all narrative variants created from a set of existing stories. Our experiments created novel, coherent and structured narratives by blending and adapting episodes from old chivalry romance pieces of work and some modern adventure videogames.

Keywords: Plot Generation, Conceptual Blending, Interactive Storytelling.

1 Introduction

Human creativity has been described as the process of producing something new (i.e., original, unexpected) and appropriate (i.e., useful, applicable) [31]. Although the basic definition of creativity suggests the invention of something entirely new and original, many authors follow a *reuse strategy* when designing a new idea or product. This is a very common practice in the entertainment industry, especially in films and games.

In narrative writing, it is a well-established fact that new stories often emerge as creative adaptations and combinations of old stories [1]. The idea of combining and adapting existing stories to create new narratives easily brings to our minds the possibility of using computational algorithms to automate the narrative generation process. Although research on interactive storytelling has been exploring the generation of interactive narratives since the 1970s [10][20], we are still far from having algorithms capable of creating complex and creative stories as those created by professional human authors. Many different strategies have been adopted in order to create plot generation algorithms, including automated planning [5][14][27][29], plot grammars

[2][18], and genetic algorithms [22][15]. However, only few works attempted to apply reuse strategies to blend existing stories in order to generate new narratives.

We call *computational narrative blending* the process in which two or more narratives are combined to generate a new narrative variant using computers. This concept comes from the notion of *conceptual blending* [6][7], which has been proposed as a fundamental cognitive process where two or more conceptual spaces are merged to form a new blended space. The blended space is partially structured by the input spaces, but it also exhibits some emergent structure of its own.

Although conceptual blending can be considered a powerful model for creativity and analysis, there are many challenges related to the application of blending in computational systems. Even though some previous works have already applied conceptual blending to computational systems [8][19][21][23][33][34], there still is no general formula on how to construct algorithmic solutions for all types of blends. In addition, the narrative domain presents extra challenges for the blending process, especially when we consider the coherence, diversity, and quality of the generated stories (see [30] for a discussion in narratology). In this paper, we have no intention to discuss theoretical questions about conceptual blending or propose a general computational model for narrative blending. Instead, we simply draw and apply ideas from the literature on conceptual blending.

This paper, inspired by conceptual blending models and considering plot generation as a plan-generation problem, proposes a new plot generation method that reuses existing stories. By combining a plan validation algorithm with a basic narrative structure, our method guarantees the logical coherence and general structure of the generated narratives.

The paper is organized as follows. Section 2 reviews related works. Section 3 introduces the concept of narrative structure and presents the specialized grail-based structure. Section 4 describes the proposed narrative blending method. Section 5 presents the results generated by our method. Section 7 offers concluding remarks.

2 Related Work

Reuse strategies have been used even in the earliest narrative generation systems. In addition, some applications of conceptual blending to narrative domains have been proposed recently. Both approaches are reviewed in this section.

One of the earliest plot generation systems to adopt *reuse strategies* is Minstrel [32]. By retrieving and transforming existing scenes stored in a special memory (called episodic memory), the system can generate new stories. For the adaptation process, Minstrel identifies similar concepts in the episodic memory and uses them to create novel scenes. Although Minstrel can produce new narratives, some of the adopted heuristics only work well for specific scenes, whereas in some cases they can result in inconsistent narratives [25].

Another system based on a reuse strategy is MEXICA [24]. The system uses a set of existing stories to build structures in memory representing content and rhetorical knowledge. During the story generation process, MEXICA retrieves from memory all

possible actions that can be performed in the current story-world state. After filtering the actions that do not satisfy a group of constraints, one of the remainder actions is selected at random as the next action in the story [25]. The MEXICA reuse strategy restricts the set of stories used to construct the rhetorical knowledge of the system to variants of the same narrative.

A more recent approach to reuse strategies is explored by Lima et al. [13][17]. The authors propose that story variants are the consequence of type interactions, which they characterize in terms of semiotic relations expressing connection, similarity, unfolding, and opposition. By applying these semiotic relations over a library of narratives of related types, their system can generate new story variants.

Inspired by the notion of *conceptual blending*, Li et al. [12] describe two systems to construct blends in a goal-driven and context-driven manner. While the first system aims at breaking the static configurations of story worlds by creating new types of objects (called gadgets) and introducing them into the narratives [11], the second has the objective of selecting a real-world object to represent an object from a fantasy world, as required in children's pretend play [36]. Although their method can introduce new objects into a narrative, their inputs for the blending process do not include any temporal dimension, which is important when blending sequences of events extracted from different narratives.

Narrative blending is also explored by Permar and Magerko [26], by way of a computational model based on conceptual blending that is capable of using familiar scripts to generate new blended scripts. In their model, narrative scripts are represented by directed acyclic graphs, where each node defines an event. As acknowledged by the authors, one of the main limitations of their method is the fact that it cannot guarantee the logical coherence of the generated scripts, which is essential for a narrative.

Although reuse strategies and conceptual blending for plot generation have been explored in previous works, none of them combines all characteristics of our method, especially its effective ability to create novel, coherent and structured narratives by blending and adapting episodes (sequences of events) extracted from different stories.

3 The Narrative Structure

The proposed plot generation method reuses episodes of existing stories that share the same narrative structure to compose new narrative variants. We use the term "narrative structure" to refer to the order and manner along which a plot evolves. This structure may be presented as a pattern (like the hero patterns proposed by Joseph Campbell [4] and Rank et al. [28]) or as a story arc [35]. In this paper, to test the flexibility of our model, instead of following a classic narrative pattern or a traditional story arc, we opted for a more specialized structure named "The Fall and Rise of the Grail Hero" [16]. This structure (here called *grail-based narrative*) encompasses old chivalry romance works and some modern adventure videogames.

In the grail-based narrative structure, the protagonist begins as a naïve person, learns about himself along successive stages, *falls down* nevertheless in a crucial instant, but is then led to rise again and move towards a high position that nobody else

could attain. As explained by Lima et al. [16], this structure is inspired on a 12th century romance of chivalry entitled *Le Conte du Graal* (*Perceval*), by the French poet Chrétien de Troyes [9], whereby the Grail literary tradition was inaugurated. The original romance of Chrétien is considered an unfinished work, as he died before completing his story. Later, four so-called *Perceval Continuations* were appended to Chrétien's text [3].

The basic grail-based narrative structure can be summarized into nine different episodes: (1) *Preparation 1* (the hero learns some skills); (2) *Failed mediation* (current skills are not enough to face challenges); (3) *Apotheosis 1* (the hero joins a community and becomes a highly reputed member); (4) *Mediation* (there is a summons to pursue the mission previously not understood); (5) *Errance* (still lacking an indispensable skill, the hero wanders in vain); (6) *Preparation 2* (meeting an old sage, the hero receives the missing instruction); (7) *Quest* (the quest effectively begins, with ample chance of success); (8) *Apotheosis 2* (the quest is finally achieved and the hero is rewarded); and (9) *Denouement* (the hero is allowed to live his new and changed live).

By analyzing the narrative of videogames, Lima et al. [16] identified the grail-based structure in the narratives of five well-known games: *The Legend of Zelda: A Link to the Past* (Nintendo, 1991), *The Legend of Zelda: Ocarina of Time* (Nintendo, 1998), and *The Witcher 3: Wild Hunt* (CD Projekt, 2015). Therefore, in this work, we considered the original romance *Le Conte du Graal* (*Perceval*), the *Continuations*, and the narratives of these games to define a *domain library*, which the proposed blending method uses to create new narrative variants.

4 Narrative Blending

The proposed narrative blending method explores the existence of a set of narratives sharing the same structure to create new variants that combine episodes extracted from different narratives, which are kept in a *domain library*. The linear structures of these narratives are also extended towards a *branching network* (as described in section 4.2), where each node is an episode containing several variants. The domain library is manually constructed by a human author and it comprises – besides the narratives' events organized by episodes – a set of *planning operators* describing the preconditions and effects of all types of narrative events used in the domain library.

In terms of conceptual blending, the domain library and the branching network of the narrative structure define the *generic space* (GS), which represents the conceptual structure that is shared by the input spaces (IS_1 and IS_2). The *input space 1* (IS_1) comprises a complete narrative variant with an episode selected to be retold, and the *input space 2* (IS_2) consists of an episode variant of the episode selected in IS_1. The input concepts IS_1 and IS_2 are induced by the generalization GS. The *blended space* (BS) comprises a new narrative variant composed through a blend of both input spaces. The *blending process* consists of an attempt – that not always succeeds – to adapt the sequence of events of IS_2 so it can replace the original events of the selected episode in IS_1 without violating the logical coherence of the narrative.

Fig. 1 illustrates the conceptual spaces for a case where IS_1 comprises a narrative variant with 4 episodes (Ep_1, Ep_2, Ep_3, Ep_4). In this space, Ep_4 is the episode selected to be retold. In this GS example, the episode Ep_4 has two variants ($Ep_{4,1}$ and $Ep_{4,2}$). While in IS_1 the episode Ep_4 is based on $Ep_{4,2}$, IS_2 comprises the alternative episode variant of Ep_4 (i.e., $Ep_{4,1}$). When both input spaces are projected into BS, the logical coherence of all events of $Ep_{4,1}$ are verified according to the preconditions and effects of their respective operators defined in the generic space. In this process, the events of $Ep_{4,1}$ are adapted to conciliate with the previous events of the story, so that characters and objects of IS_1 can replace characters and objects that share similar roles in the event sequence of $Ep_{4,1}$.

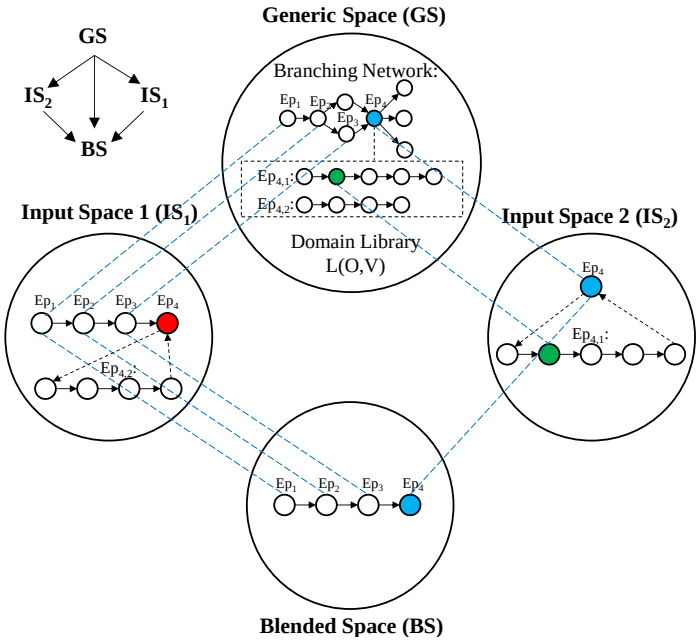

Fig. 1. Conceptual spaces.

4.1 Basic Definitions

In the proposed method, an *event* is denoted by an atomic formula of the form $T(t_1, \ldots, t_n)$, where T defines the type of the event (e.g., fight, kill, save) and the terms t_n (also called *parameters*) represent the elements involved in the event (e.g., characters, places, objects). For example, `go(Perceval, Waste Forest, Arthur's court)` represents an event where *Perceval* (a character) goes from *Waste Forest* (a place) to *Arthur's court* (another place).

An event is an instance of an *operator*, which establishes all restrictions (e.g., temporal, spatial) for the occurrence of the event (*preconditions*) and the effects that result from the occurrence of the event (*postconditions*). More specifically, an *operator* is a triple $o_i = (name_i, preconditions_i, postconditions_i)$, where $name_i$ is an atom-

ic formula with variables, and both $preconditions_i$ and $postcondions_i$ are sets of literals (i.e. positive or negative atoms). For example, the operator for the `go` event is defined by: **o1** = (**name1** = `go(CH, PL1, PL2)`, **preconditions1** = `character(CH), place(PL1), place(PL2), at(CH,PL1)`, **postconditions1** = `¬at(CH, PL1), at(CH, PL2))`, where ¬ is the negation symbol.

Any episode of the branching network is a sequence of events, which are consistent with all restrictions established by the operators. This sequence of events, also called a *plan P*, is associated with a text label, called the *episode name* (*ep*). Therefore, an *episode Ep* is the pair (ep, P). A *narrative N* is a sequence of episodes: $N = \langle Ep_1, Ep_2, \cdots, Ep_k, \cdots \rangle$. In this context, a branching network node may have several *episode variants*. Each pair (ep_k, P_k) in the narrative represents an *episode variant*, where ep_k identifies the name of the episode and P_k describes a way of telling ep_k.

For instance, the following event sequence defines the episode Ep_6 (Preparation 2) of the narrative of Chrétien's *Perceval*: `go(Perceval, Arthur's land, hermitage), meet(Perceval, hermit, hermitage), tell(hermit, Perceval, religious knowledge).`

A *story variant* is a quadruple $V_i = (n_i, N_i, \omega_i, f_i^0)$, where n_i is the *narrative name*, N_i is a narrative (called *narrative variant*), ω_i is a *list of symbols* with all names that are used to describe the events and states of V_i, and f_i^0 is a set of facts describing the initial state of the story. A *fact* is an assertion about an entity of the narrative variant (e.g., character, object, place), which can be the assignment of a role to an entity (e.g., `hero(Perceval), place(Arthur's court), object(Red Knight's armor)`), or the assertion about the attributes of an entity (e.g., `alive(Perceval), threatened(Hyrule), defeated(Red Knight)`), or the existence of a relationship between entities (e.g., `at(Perceval, Arthur's court), has(Red Knight, Red Knight's armor), love(Perceval, Blancheflor)`). The set of facts holding at a given instant of time constitutes a *state*. The state at the beginning of a story is called *initial state*. At any point of the narrative, we can determine what is the current state, because we have the operators. We define $S_k(V_i)$ as the state of V_i before episode Ep_k of N_i. In this case, $S_1(V_i) = f_i^0$.

The *domain library* is a pair $L = (O, V)$, where $O = \{o_1, o_2, \cdots, o_w\}$ is a set of operators and $V = \{V_1, V_2, \ldots, V_n\}$ is a set of story variants. In our system, the library is specified in an XML file (an example of domain library is available in a separated online document: http://www.icad.puc-rio.br/~logtell/narrative-blending/narrative_blending_db.xml). When the domain library is loaded by the system, the list of symbols $\omega_i \in V_i$ is automatically created and filled with all unique values found in the parameters of the story events and initial state. Each story variant V_i is associated with its own list of symbols, which provides access to all names (characters, places, objects) that were used to describe the story.

4.2 Branching Network

The first step of the narrative blending method involves the construction of a branching network structure by combining the linear sequences of episodes of the stories defined in domain library. The *branching network* is modeled as a directed graph $G =$

(A, E), where A is a set of episode labels (graph nodes), and E is a set of 2-element subsets of A, called *episode edges*.

In order to create the branching network, the sequences of episode labels of all stories of the domain library are extracted. Then, two general border events (called *begin* and *end*) are added to each sequence and grouped as a network structure, in which each episode becomes a node label (Fig. 2a). In the next step, the algorithm combines nodes (through unification) with the same episode labels to transform the initial network into a branching network (Fig. 2b), in a process we name "fusion by equality."

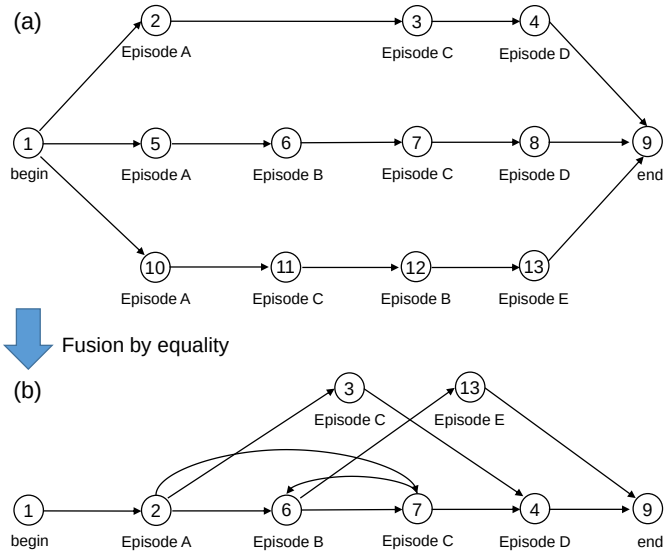

Fig. 2. Example of episode unification transforming the initial network (a) into the branching network (b).

By creating the branching network according to the domain library, the original narrative structure is extended and enriched with alternative crossing paths extracted from stories that follow a similar structure. Fig. 3 illustrates the branching network automatically created for a domain library composed of 3 grail-based stories: Chrétien's *Perceval* complemented by the *Second Continuation; The Legend of Zelda: A Link to the Past*; and *The Legend of Zelda: Ocarina of Time*. In this network, both branches α and γ were stablished due to an inversion in the order of the episodes (4) and (5) in *The Legend of Zelda: Ocarina of Time*, and branch β was added due to the fact that *The Legend of Zelda: A Link to the Past* lacks episode (5). In addition, a new episode – (9) Magical Agent – was added to the branching network to describe an episode of *The Legend of Zelda: A Link to the Past* where Link gets a magical weapon (the Silver Arrow), which is the only weapon capable of defeating the evil Ganon.

A sequence of episodes extracted from the branching network is a *graph walk*. The alternative walks that can be extracted from the branching network – without repeating episodes – represent the different orders in which the episodes of the narrative structure can be arranged to create new story variants.

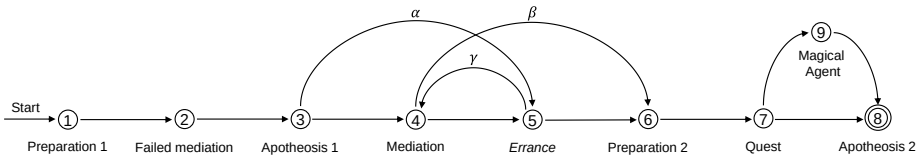

Fig. 3. Branching network created for three grail-based variants.

4.3 Episode Blending

After creating the branching network for the narrative structure, the next step involves the generation of all possible story variants that can be created by combining event sequences of different episodes extracted from the narratives of the domain library. This is done through successive applications of a process that we called *episode blending*. This process adapts a single sequence of events (an episode) extracted from a narrative of the domain library to use it in another narrative. The process comprises two tasks: *adaptation* and *validation*. While the sequence of events is being adapted to fit into a different narrative, the logical coherence of the events is constantly being checked to guarantee that they are not violating the coherence of the narrative.

Given two input spaces (IS_1 and IS_2), a generic space (GS), and a blended space (BS), the episode blending process comprises the following steps:

1. Convert the sequence of events of IS_2 into a *generic plan* $GP = \{e_1, \ldots, e_n\}$, where the parameters of each event e_i are replaced by uninitialized *variables*;
2. Select the next event e_i of GP (initially, the first event), and assign new values to the uninitialized/failure *variables* of the event according to the list of symbols of the narrative variant $V_1 \in IS_1$;
3. Apply the preconditions of the event operator $O_i \in GS$ to *validate* the logical coherence of e_i:
   a. If the validation of e_i *succeeds*, apply the postconditions (effects) of O_i over the current state of the narrative variant $V_1 \in IS_1$, then add e_i to BS, and then jump back to step 2 in order to proceed to the next event of GP;
   b. If the validation of e_i *fails*, identify the variable that caused the logical *failure*:
      (1) If the variable that caused the failure was initialized in e_i, then jump back to step 2 in order to try the next possible value for the variable;
      (2) If the variable that caused the failure was initialized in an *event prior to* e_i, then backtrack the process to the step where the variable was initially defined (reverting all changes made to the current state and removing all events added to BS). Continue from step 2 in order to try the next possible values for the variable;
      (3) If the variable that caused the failure was initialized in a *previous episode* (considering that the episode blending process is being applied to more than one episode of V_1), then backtrack the whole process to the episode blending operation where the variable was initially defined (reverting all changes made to the current state and removing all events added to the episode variant that is being created). Continue from step 2 in the previous instance of the episode blending process to try the next possible value for the variable.

In order to exemplify the episode blending process, let us consider a case where *input space 1* (IS_1) comprises an ongoing story variant V_1 with all episodes and events logically validated. Also, suppose that episode Ep_6 (*Preparation 2* in the narrative of Chrétien's *Perceval* complemented by the *Second Continuation*, see section 4.1) is the episode selected to be retold. The *current state* of V_1 before episode 6 comprises some facts, such as: `hero(Perceval), alive(Perceval), ... .`

In this example, *input space 2* (IS_2) contains the following alternative variant of episode 6 extracted from the narrative variant *The Legend of Zelda: A Link to the Past*:[1] `go(Link, Dark World, Dark Palace), fight(Link, Helmasaur King, Dark Palace), defeat(Link, Helmasaur King, Dark Palace), rescue(Link, First Maiden, Dark Palace).`

In the *generic space*, the operators related to *go*, *fight*, *defeat*, and *rescue* are totally relevant. As an example, the operator of the event *go* is $\mathbf{o_1}$ (see section 4.1). In order to use the sequence of events of IS_2 to replace episode 6 in $V_1 \in IS_1$, the events of IS_2 must be adapted and logically validated according to the current state of V_1.

The first step of the adaptation and validation process consists in converting the sequence of events of IS_2 into a *generic plan* that uses *variables* to represent the parameters of the events. Therefore, the sequence of events of IS_2 becomes: `go(A, B, C), fight(A, D, C), defeat(A, D, C), rescue(A, E, C).`

Starting from the first event of the generic plan, the algorithm replaces the variables of the event with symbols extracted from the list symbols $\omega_1 \in V_1$: $\omega_1 =$ {`Perceval, Arthur's land, hermit, hermitage, Red Knight, Lord of the Horn, mighty castle, Fair Unknown, open forest, beautiful woman, ...`}. The initial values assigned to the variables used in the first event of the generic plan are: `A = Perceval, B = Arthur's land, C = hermit`.

After instantiating the event, the algorithm uses the preconditions of the event operator to validate its logical coherence. The validation process iterates through all predicates of the precondition checking the validity of them according to the current state of $V_1 \in IS_1$. In the example, the preconditions of the operator o_1 become: `character(Perceval), alive(Perceval), place(Arthur's land), **place(hermit)**, at(Perceval, Arthur's land).`

Analyzing the *current state* of V_1, we see that all facts are true, but **`place(hermit)`** is not (given that `hermit` is not a place). Therefore, the assignment fails, and `C` is identified as the variable that caused the failure. Then, the algorithm backtracks and tries the next possible value for `C` according to the list of symbols, which is `Red Knight`. However, the same logical failure will happen again (`Red Knight` is not a place as well). All preconditions will be true only when `mighty castle` is assigned to `C`. Then, we have `A = Perceval, B = Arthur's land, C = mighty castle.` The event is thus instantiated and added to the current episode variant: `go(Perceval, Arthur's land, mighty castle).`

When the validation of an event succeeds, the postconditions (effects) of the operator are applied over the current state. Considering the postconditions defined by o_1,

[1] This is a simplified version of the episode that was created to illustrate the blending process (the original episode comprises the rescue of seven maidens).

the fact `at(Perceval, Arthur's land)` will be removed from the current state of V_1 and `at(Perceval, mighty castle)` will be added to the current state of V_1.

The algorithm applies the same procedure to the subsequent two events of the generic plan (`fight(A, D, C)` and `defeat(A, D, C)`), where some variables are already assigned, and others are not. The result is the following episode variant: `go(Perceval, Arthur's land, mighty castle)`, `fight(Perceval, Lord of the Horn, mighty castle)`, `defeat(Perceval, Lord of the Horn, mighty castle)`. Also, the postconditions of the operators add some facts and remove others.

The last event of the generic plan is `rescue(A, E, C)`, where `A` and `C` are already assigned. However, all attempts of assigning a value for `E` will fail. When all assignment options for a variable are unsuccessful, the algorithm selects – among the variables that are being used in the current event – the one that is closely related with the precondition that caused the failure, giving priority to the variables that were recently successfully assigned. This variable is then identified as the variable that is causing the logical failure. In the example, the variable C affects one term in the precondition of the operator *rescue* that failed.

After identifying the variable that is causing the logical failure, the algorithm backtracks to the recursive call where variable `C` was initially defined and continues the process of trying the remaining values that can be assigned to `C`. The backtracking process reverts all changes made to the current state and removes all events added to the episode variant that is being created.

As variable `C` was defined in the first event of the episode variant, the algorithm returns to its starting point. The *go* operator fails `C = Fair Unknown`, but succeeds with `C = open forest`. Then, the event is instantiated and added to the current episode variant: `go(Perceval, Arthur's land, open forest)`

The algorithm repeats the same steps until the last event is `rescue(A, E, C)`, where `A` and `C` are already assigned. However, this time, the preconditions of the `rescue` operator will succeed when the algorithm gives `beautiful woman` to `E`. At this point, the final plan that replaces the original plan of episode 6 is: `go(Perceval, Arthur's land, open forest)`, `fight(Perceval, Fair Unknown, open forest)`, `defeat(Perceval, Fair Unknown, open forest)`, `rescue(Perceval, beautiful woman, open forest)`.

This is one of the variants. However, given a domain library L and a branching network G, a recursive function easily generates all possible narrative variants. This function can be found in the repository of our project: http://www.icad.puc-rio.br/~logtell/narrative-blending/narrative_blending_algorithm.pdf.

5 Application and Results

To apply and evaluate the proposed method, we implemented in C# a tool to assist amateur/professional writers to visualize all narrative variants created from a set of existing stories (Fig. 4). Besides applying the proposed narrative blending method, the tool also permits users to interfere in the initial state of the narratives by changing existing facts and adding new ones. This feature helps the generation of more personalized narratives (for example, allowing the author to decide who will play the role of

hero in the stories). Users can define the customized initial state by manually writing logical facts or by using a *state design tool* that allows them to create characters, objects, and establish relations between them visually.

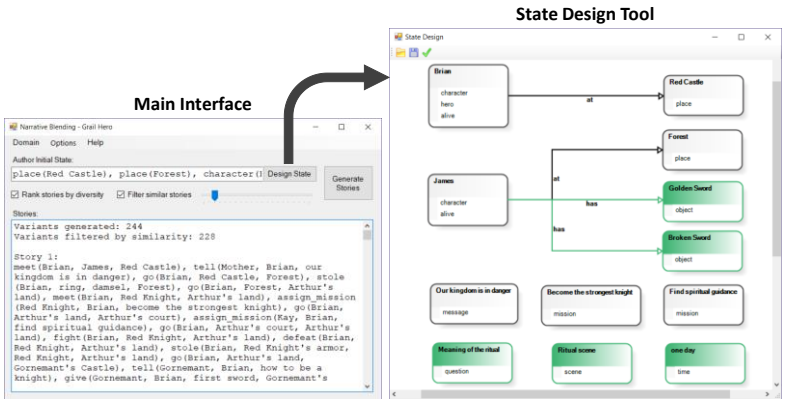

Fig. 4. User interface of the narrative blending tool.

To demonstrate the capacity of our method to generate coherent and diversified narratives, we present below a variant that emerged from the following three different narratives: *The Legend of Zelda: A Link to the Past* (events highlighted in *italic*), *The Legend of Zelda: Ocarina of Time* (events highlighted in **bold**), and Chrétien's *Perceval* complemented by the *Second Continuation* (events highlighted with underline)). The final text (in the form of predicates) is:

tell(James, Brian, our kingdom is in danger), assign_mission(James, Brian, become the strongest knight), go(Brian, Red Castle, Forest), fight(Brian, Armos Knights, Forest), defeat(Brian, Armos Knights, Forest), get(Brian, Pendant of Courage, Forest), go(Brian, Forest, Desert Palace), fight(Brian, Lanmolas, Desert Palace), defeat(Brian, Lanmolas, Desert Palace), get(Brian, Pendant of Power, Desert Palace), go(Brian, Desert Palace, Tower of Hera), fight(Brian, Moldorm, Tower of Hera), defeat(Brian, Moldorm, Tower of Hera), get(Brian, Pendant of Wisdom, Tower of Hera), go(Brian, Tower of Hera, Lost Woods), get(Brian, Master Sword, Lost Woods), join(Brian, Knights of Hyrule), go(Brian, Lost Woods, Forest), meet(Brian, James, Forest), go with(Brian, James, Forest, Hyrule Castle), give(James, Brian, Broken Sword, Hyrule Castle), watch(Brian, ritual scene, Hyrule Castle), fail to ask(Brian, James, meaning of the ritual, Hyrule Castle), sleep(Brian, one day), *assign_mission(Sahasrahla, Brian, find spiritual guidance),* wander(Brian, seven years), **go(Brian, Hyrule Castle, Ice Palace), awake(Brian, Fifth Maiden, Ice Palace), go(Brian, Ice Palace, Gargoyle's Domain), awake(Brian, Fourth Maiden, Gargoyle's Domain), go(Brian, Gargoyle's Domain, Dark Palace), awake(Brian, First Maiden, Dark Palace), go(Brian, Dark Palace, Skull Woods), awake(Brian, Third Maiden, Skull Woods), go(Brian, Skull Woods, Turtle Rock), awake(Brian, Seventh Maiden, Turtle Rock), go(Brian, Turtle Rock, Swamp Palace), awake(Brian,**

Second Maiden, Swamp Palace), *go(Brian, Swamp Palace, Ganon's Tower Entrance)*, *open(Brian, Ganon's Tower, Ganon's Tower Entrance)*, go(Brian, Ganon's Tower Entrance, Hyrule Castle), watch(Brian, ritual scene, Hyrule Castle), ask(Brian, James, meaning of the ritual, Hyrule Castle), give(James, Brian, Golden Sword, Hyrule Castle), repair(Brian, Golden Sword, Hyrule Castle), crown(James, Brian, Hyrule Castle).

In terms of computational performance, the proposed method has limitations. As exhaustive search is the strategy adopted, the algorithm systematically enumerates and checks all possible ways in which the episodes and events of the narratives can be combined. Consequently, the complexity of the algorithm grows according to the numbers of variants, episodes, events and also the number of facts and symbols used to describe the initial state of the variants. For example, a domain library with four narratives and 6 different walks takes 15.4 min to generate all possible variants (in a computer with an Intel Core i7-7820HK, 2.90 GHZ, and 16 GB of RAM). In this test, the original narratives were: *Perceval* complemented with the *Second Continuation* (8 episodes, 87 events, 144 facts in the initial state, and 65 different symbols), *The Legend of Zelda: A Link to the Past* (6 episodes, 61 events, 81 facts in the initial state, and 50 different symbols), *The Legend of Zelda: Ocarina of Time* (8 episodes, 43 events, 64 facts in the initial state, and 43 different symbols), and *The Witcher 3: Wild Hunt* (8 episodes, 67 events, 75 facts in the initial state, and 49 different symbols).

6 Concluding remarks

In this paper we presented a new plot generation method that reuses fragments of existing stories to compose new narrative variants. Inspired by the notion of conceptual blending (especially [7] and [8]), our method adapts sequences of events extracted from different narratives in a way that they can fit into a new narrative variant without violating the logical coherence of the story. The combination of plan validation with an exhaustive search strategy, allows the generation of coherent and diversified plots. In addition, all plots are constructed according to a specific narrative structure, which guarantees the overall dramatic structure of the generated stories.

Although the proposed narrative blending method can generate coherent narratives even when episodes of distinct narratives are combined, the process to generate all possible variants for a domain library is a computationally expensive task even when a small number of narratives are considered. In addition, it is important to point out that our method is far from being able to perform creative blending tasks at the same level that talented human authors would do. The use of plan validation is an excellent way of guaranteeing the logical coherence of generated narrative, but on the other hand strict logic sometimes causes some interesting episodes to be discarded as result of very small coherence violations that could easily be fixed by a human author.

Many future works are envisaged, especially regarding the optimization of the blending process. Since our method to solve the problem of finding all variants for a domain library is based on a recursive algorithm – where each episode blend task can be considered a sub-problem – and given the fact that many of the episode blends

involve similar or equal inputs, one can easily imagine an optimized solution using a dynamic programming strategy. By using memoization to store the results of previous blends and returning the cached results when necessary, the performance of the whole narrative blending process would likely be improved.

Acknowledgements. We would like to thank CNPq (National Council for Scientific and Technological Development) and FINEP (Brazilian Innovation Agency), which belong to the Ministry of Science, Technology, and Innovation, for the financial support.

References

1. Barthes, R.: Theory of the Text. In: Young, J. C. (ed.) Untying the Text: a Post-Structuralist Reader. Routledge & Kegan Paul, Oxfordshire, England (1981).
2. Bringsjord, S., Ferrucci, D. A.: Artificial Intelligence and Literary Creativity: Inside the Mind of BRUTUS, a Storytelling Machine. Psychology Press, Sussex, England (1999).
3. Bryant, N. (trans.).: The Complete Story of the Grail. D.S. Brewer, England (2015).
4. Campbell. J.: The Hero with a Thousand Faces. Princeton University Press, Princeton, United States (1973).
5. Ciarlini, A. E. M., Pozzer, C. T., Furtado, A. L., Feijó, B.: A logic-based tool for interactive generation and dramatization of stories. In: Proceedings of the International Conference on Advances in Computer Entertainment Technology, pp. 133–140 (2005).
6. Fauconnier, G., Turner, M.: Conceptual Integration Networks. Cognitive Science 22 (2), 133–187 (1998).
7. Fauconnier, G., Turner, M.: The way we think: Conceptual blending and the mind's hidden complexities. Basic Books, New York, United States (2003).
8. Goguen, J.: Mathematical models of cognitive space and time. In: Andler, D., Ogawa, Y., Okada, M., Watanabe, S. (eds.) Reasoning and Cognition: Proceedings of the Interdisciplinary Conference on Reasoning and Cognition, pp. 125–128 (2006).
9. Kibler, W. W. (trans.): Chrétien de Troyes - Arthurian Romances. Penguin Books, London, England (1991).
10. Klein, S., Aeschlimann, J. F., Balsiger, D. F., Converse, S. L., Court, C., Forster, M., Lao, R., Oakley, J. D., Smith, J.: Automatic Novel Writing: A Status Report. Technical Report 186, Computer Sciences Department, University of Wisconsin, Madison (1973).
11. Li, B., Riedl, M. O.: A Phone That Cures Your Flu: Generating Imaginary Gadgets in Fictions with Planning and Analogies. In: Proceedings of the 4th Workshop of Intelligent Narrative Technologies, pp. 41–48 (2011).
12. Li, B., Zook, A., Davis, N., Riedl, M. O.: Goal-Driven Conceptual Blending: A Computational Approach for Creativity. In: Proceedings of the 2012 International Conference on Computational Creativity, pp. 9–16 (2012).
13. Lima, E. S., Feijó, B., Casanova, M. A., Furtado, A. L.: Storytelling Variants Based on Semiotic Relations. Entertainment Computing 17, 31–44 (2016).
14. Lima, E. S., Feijó, B., Furtado, A. L.: Hierarchical Generation of Dynamic and Nondeterministic Quests in Games. In: Proceedings of the International Conference on Advances in Computer Entertainment Technology, Article 24, pp. 1–10 (2014).
15. Lima, E. S., Feijó, B., Furtado, A. L.: Procedural Generation of Quests for Games Using Genetic Algorithms and Automated Planning. In: Proceedings of the XVIII Brazilian Symposium on Computer Games and Digital Entertainment, pp. 495–504 (2019).

16. Lima, E. S., Furtado, A. L., Feijó, B., Casanova, M. A.: Towards Reactive Failure-Recovery Gameplaying: The Fall and Rise of the Grail Hero. Proceedings of the XV Brazilian Symposium on Computer Games and Digital Entertainment, pp. 262–271 (2016).
17. Lima, E. S., Furtado, A. L., Feijó, B.: Storytelling Variants: The Case of Little Red Riding Hood. In: Proceedings of the 14th International Conference on Entertainment Computing, Trondheim, Norway, pp. 286–300 (2015).
18. Machado, I., Paiva, A., Brna, P.: Real characters in virtual stories: Promoting interactive story-creation activities. In Proceedings of the 1st International Conference on Virtual Storytelling, pp. 127–134 (2001).
19. Martinez, M., Besold, T., Abdel-Fattah, A. M. H., Kuehnberger, K. U., Gust, H., Schmidt, M., Krumnack, U.: Towards a Domain-Independent Computational Framework for Theory Blending. In: AAAI Fall Symposium: Adv. in Cognitive Systems, pp. 210–217 (2011).
20. Meehan, J.: TALE-SPIN, an interactive program that writes stories. In: Proceedings of the Fifth Interactional Joint Conference on Artificial Intelligence, pp. 91–98 (1977).
21. O'Donoghue, D., Abgaz, Y., Hurley, D., Ronzano, F.: Stimulating and Simulating Creativity with Dr Inventor. In: Proceedings of the Sixth International Conference on Computational Creativity, pp. 220–227 (2015).
22. Ong, T., Leggett, J. J.: A genetic algorithm approach to interactive narrative generation. In Proceedings of the 15th ACM Conference on Hypertext Hypermedia, pp. 181–182 (2004).
23. Pereira, F. C., Cardoso, A.: Optimality principles for conceptual blending: A first computational approach. AISB Journal 1 (4), 351-369 (2003).
24. Pérez y Pérez, R., Sharples, M.: MEXICA: A computer model of a cognitive account of creative writing. Experimental and Theoretical Artificial Intelligence 13, 119–139 (2001).
25. Pérez y Pérez, R., Sharples, M.: Three computer-based models of storytelling: BRUTUS, MINSTREL and MEXICA. Knowledge-Based Systems 17, 15–29 (2004).
26. Permar, J., Magerko, B.: A conceptual blending approach to the generation of cognitive scripts for interactive narrative. In: Proc. of the 9th AIIDE Conference, pp. 44–50 (2013).
27. Pizzi, D., Cavazza, M.: Affective Storytelling Based on Characters' Feelings. In: AAAI Fall Symposium on Intelligent Narrative Technologies, pp. 110–117 (2007).
28. Rank, O., Raglan, L., Dundes, A.: In Quest of the Hero. Princeton University Press, Princeton, United States (1990).
29. Riedl, M. O., Young, M.: Narrative Planning: Balancing Plot and Character. Journal of Artificial Intelligence Research 39, 217–26 (2010).
30. Schneider, R., Hartner, M. (eds.): Blending and the Study of Narrative: Approaches and Applications. De Gruyter, Berlin, 2012.
31. Sternberg, R. J.: Handbook of Creativity. 1st edition. Cambridge University Press, Cambridge, England (1998).
32. Turner, S.R.: MINSTREL: A computer model of creativity and storytelling. PhD Thesis, Computer Science Department, University of California (1993).
33. Veale, T. O'Donoghue, D.: Computation and Blending. Cognitive Linguistics 11 (3/4), 253-281 (2000).
34. Veale, T.: How to Blend Concepts and Influence People: Computational Models of Conceptual Integration. Theoria et Historia Scientiarum 6 (1), 363–398 (2002).
35. Yorke, J.: Into The Woods: How Stories Work and Why We Tell Them. Penguin Books, London, England (2014).
36. Zook, A. E., Riedl, M. O., Magerko, B. S.: Understanding Human Creativity for Computational Play. In: Proceedings of the Second International Conference on Computational Creativity, pp. 42–47 (2011).