Towards a Computational Model of Dramatic Tension*

Nicolas Szilas and Urs Richle

TECFA, Faculté de Psychologie et des sciences de l'éducation
University of Geneva
Genève, Switzerland
Nicolas.Szilas@unige.ch

Abstract

One of the approaches to generate narrative consists in modeling narrative in terms of a deep structure, as introduced by narrative theories in the middle of the 20th century.

This papers revisits this computational approach, and raises the central issue of dramatic tension: Would it be possible to build a computational model of dramatic tension, where tension could be managed according to the well known ascending/descending dramatic curve?

The paper describes a new computational model of narrative, based on a set of structural narrative elements (goals, tasks, obstacles, side-effects), a hierarchical and modular approach, a paradox-based model of dramatic tension and a solution for managing endings.

The papers illustrates this theoretical model with a full example.

1998 ACM Subject Classification I.2 Artificial Intelligence

Keywords and phrases computational model of narrative, dramatic tension, structural writing, structuralism, narrative theories.

Digital Object Identifier 10.4230/OASIcs.CMN.2013.257

1 Structures and Paradox

There exists a multitude of computational models dedicated to the narrative generation. A promising approach is based on structuralist theory. The principle is rather simple: starting from Lévi-Strauss study of myths [23], the same way a unique deep structure of the myth can produce many different specific myths across communities and cultures, a unique core structure of a narrative could produce, when simulated, a large variety of different stories, across different executions. The key feature of such structures – and this is important to insist since the term "structure" shares different and sometimes contradictory meanings [38] – is their *atemporal* nature [23]. Such deep structures contain various interconnected narrative elements that, at least in part, do not specify a *specific* temporal relation between these elements. For example, stating that two characters are in conflict or that the narrative revolves around two thematic elements are atemporal specifications. These specifications later produce temporally ordered actions or events.

There has been some attempts to build systems based on these principles. In Black Sheep [20], the six actant model of Greimas [17] was implemented, but the model remained very basic, so the benefit in terms of variations could not be observed. In *IDtension* [36, 34], a specific structural model was developed for Interactive Drama, integrating the dramatic

* This research is supported in part by a grant from the Swiss National Science Foundation (J. Dumas and N. Szilas, principal investigators).

Workshop on Computational Models of Narrative 2013.
Editors: Mark A. Finlayson, Bernhard Fisseni, Benedikt Löwe, and Jan Christoph Meister; pp. 257–276

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

notion of ethical conflict. Although a full scale and playable story could be implemented [18], it appeared quite difficult to write deep structures, and these structures were often designed according to branching or conditional approaches.

A deep narrative structure is a compact set of narrative elements that describes the core issue in a narrative. This structure is then processed to produce narrative events. Such a processing is possible if two components are implemented:

- An "Action Generator", able to generate possible actions, based on a deep structure.
- A "Narrative Sequencer", able to select actions among the many ones that are generated by the Action Generator, based on how the narrative is perceived by the audience. This is typically implemented with a model of the user [2, 35, 42].

This Narrative Sequencer is very challenging to build, because it is hard to find an agreement within narrative theories regarding the set of criteria that determine that a sequence of events is perceived as a narrative or not. Which criteria should be considered first, to guide the narrative generation? Closure, conflict, emotions, etc.? We suggest to consider *dramatic tension* as a good candidate for the model of the user, because it is widely recognized and intuitively observed that in a drama, tension increases and decreases during the narrative. However, it is not straightforward to implement the concept of dramatic tension, because this concept is not well defined in narrative theories. Modeling tension consists in evaluating the tension of generated actions, which implies the construction of a computational model of the dramatic tension. This is different from tagging events with a tension level, as in the interactive drama *Façade* [25]. The tagging approach only works when narrative events are not generated but pre-written.

In dramaturgy and screenwriting, the concept of tension is associated with the dramatic curve, that describes dramatic tension as a function of time: dramatic tension increases up to the climax, occurring at approximatively two thirds into the narrative and then decreases until the end of the narrative. This curve was introduced by G. Freytag in the mid nineteen century [24]. But what is it exactly that increases and then decreases during the typical drama? What makes the tension increase or decrease? This is usually not clearly defined [5]. There exist however other related concepts that are better defined in various narrative theories: suspense, conflict and paradox.

Suspense can be rigorously defined [8]. It occurs when a certain story state (target state) is strongly hoped by the audience and when among the various outcomes that can be anticipated, the target state is perceived to have very little chance to occur. (Reciprocally, suspense can be defined via a fear of a negative state that has a high chance to occur.) Suspense can be produced either at the story or at the discourse level.

Conflict denotes the "struggle in which the actors are engaged" [29]. For many theorists and writers of drama, this is a core concept [13, 14, 22, 26], and it is common to read "drama is conflict". This is, as tension, a very common term. The main conceptual difficulty is that there are different types of conflicts, which, when gathered into one single word, create a fuzzy concept. Generally speaking, conflict arises when the two goals cannot be reached at the same time. Depending on the kinds of goals, different kinds of conflict can be distinguished, such as conflict of interest, ethical conflict, external vs internal conflict.

Finally, dramatic tension can be the result of the paradoxical nature of narrative [28]. According to B. Nichols, any narrative is based on a paradox, that is a logical impossibility, as illustrated in classical paradoxes: "Epimenides was a Cretan who said, 'Cretans always lie'." (paradox of Epimenides). The narrative paradox is the juxtaposition of two contradictory terms, that successive narrative actions attempt to resolve. Trying to "resolve the unresolvable" along the major part of a story inevitably creates tension. In some case, the story can ends

by operating a major shift in the paradox, where the paradox's term are re-interpreted with a new point of view. In this case, the paradox is not solved but rather dissolved: it does not exist anymore. In other cases, the paradox remains until the end. This approach finds it roots in the analysis of the myth by C. Lévi-Strauss [23] and can be related to, outside the narratology realm, paradoxical injunction [41] and humor [21].

As an example, in a love story, the male hero must accomplish a "bad" activity to conquer his love, but if he performs this activity, she might learn it and refuse him. Nichols provides many examples in feature films. For example, for *The Birds*, from A. Hitchcock: "If I am to win Mitch, I must become part of his family, but if I become part of his family, I can not win Mitch".

There exist computational models of both suspense [11] and conflict [4, 6, 31, 34, 40]. In this paper, we want to explore the modeling of paradox. What is particularly interesting in this theory is that a certain configuration of narrative elements can generate an infinite temporal behavior, when characters try to solve the paradox but, by definition, cannot solve it. If the paradox could be expressed as a structural property of the above-mentioned deep narrative structures, then we would have an elegant and powerful way to generate a flow of actions that are narratively relevant, since they express the fundamental struggle of the narrative.

2 A computational model of narrative based on dramatic tension

2.1 General architecture

Based on the narratological discussion that precedes, we will sketch a computational model able to generate a flow of events that are narratively relevant, this relevance being assessed according to a modeling of the dramatic tension. This computational model will meet the three following requirements:

First, the model must be able to easily identify what is at stake in the narrative, that is the fundamental paradox of the narrative. Other elements should be implemented as well – without them, the narrative would not work – and they should serve to express the fundamental paradox.

Second, the model must be able to expand elements in the structures into a variety and quantity of narrative actions, which enables to make the deep structure work. This will be the role of the "Action Generator" mentioned above.

Third, the concept of dramatic tension must be formalized, so that 1) at each moment, it is possible to evaluate the current tension; 2) for each possible action, it is possible to associate the impact on the tension and 3) a strategy for managing this tension is designed.

While the focus of this paper is on the deep narrative structure, other components also play an essential role and will be described as well. In particular, it is important to keep in mind that the elements of the structure are not, in many cases, the actions that are performed. The latter are generated from the former by the Action Generator (see Section 2.5).

2.2 The core components of the Structural Model

The basic elements of the structural model will be largely inspired from previous research [36, 37], although alternative representations could be explored [9].

Four simple elements, along with their interrelations will constitute the basic "bricks" to build a structure and are called *nodes*:

Goals: A goal represents a state of affairs that a character may wish to reach. This concept is omnipresent in dramaturgy [22, 26, 39] and also intervenes is some recent definitions of narrative [30], since it corresponds to the fundamental notions of characters' actions and intentions. A large number of computational models of narrative implement goals [1, 4, 10, 27, 36, 43], in particular agent-based models, since goals are a fundamental concept in Artificial Intelligence.

The character who has a goal is called the *actor* of the goal. In the discrete case (default), the goal is either reached or not reached. In the continuous case, the goal is associated to an *achievement value*, that varies between 0 and 1, 1 meaning that the goal is fully reached. When a goal is activated (added in the active structure, see below), the achievement value is set to 0. Each time a task connected to the goal succeeds, the achievement value is incremented by a fixed parameter called the *increment*.

Tasks: A task represents a concrete action that a character can perform to reach a goal. It corresponds to the notion of action or operator in several AI or robotic frameworks. However, in the current model, actions are based on tasks but cannot be reduced to tasks, as explained in Section 2.5.

When performed, a task leads either to a success, in which case the goal is reached (see above), or to a failure, in which case the goal in not reached. By default, a performance of a task is successful. But obstacle attached to the task may lead to a failure.

Obstacles: An obstacle represents an event attached to a task that may hinder the reaching of a goal, when the task is attempted by a character. This is where the model departs from classical AI representations, due to the nature of narrative: while the focus in robotics is to reach a goal as efficently as possible, the focus of narrative is on the difficulties met by the actors when they try to reach their goals. Obstacle is a classical way of representing these difficulties, and is borrowed from dramaturgy and screenwriting [22, 26]. Obstacles implement the important notion of failure that is central in the narrative theory of C. Bremond [7].

An obstacle is attached to a task. If an obstacle *triggers* after an attempt to perform the task, the goal is not reached and it may impact the storyworld (see relations below). An obstacle is associated with a probability of triggering, called the *chance*. It is comprised between 0 and 1 (1 by default). This value is author-defined, according to the frequency of occurrence the author wants to give to the obstacle. For some obstacles, the chance to trigger is undecided: the decision to trigger them is at the discretion of the narrative engine. These obstacles are called *free obstacles*.

Side-effects: A side-effect represents an event attached to a task that may trigger when the task is performed by a character. But contrary to an obstacle it does not prevent the reaching of a goal. Side-effects thus implements a variant of a task. Side-effects are not a prominent concept in narrative theory or dramaturgy. However, it is included in the model because we lacked such a concept, when a creative author was writing stories with IDtension [12], which implements obstacles but not side-effects. More precisely with obstacle, plenty of variants of a task can be written, that occur unpredictably when a character starts to perform a task, but they all lead to a failure. Symmetrically, it would be relevant to write variants of tasks which occur unpredictably even when the task succeeds. When a side-effect is not known in advance by the character, it corresponds to an involuntary action, as described by C. Bremond: An agent undertakes a task but performs at the same time an "involuntary action" [7, p. 237].

As obstacles, side-effects are associated to a chance (1 by default).

While AI-oriented models of narrative use world states to express the consequences and preconditions of action execution, the proposed model avoids this kind of representation that is cumbersome for creative authors. Rather, the dynamic of goals, tasks, obstacles and side-effects is entirely described with *relations* (or *links*) between these nodes. The following relations have been identified:

Reaching: A *reaching relation* connects a task to the goal it enables to reach. In the continuous case, it contains the value added to the achievement value, called the *achievement increment*.

Attachment: An *attachment relation* connects an obstacle or a side effect to the task that may trigger it.

Sub-goaling: A *sub-goaling relation* connects an obstacle to a goal: it indicates that the obstacle (called the triggering obstacle), when it triggers, activates the goal (called the sub-goal). The task remains blocked by the obstacle until the sub-goal is reached. In the continuous case, the task remains blocked by the obstacle until the achievement value of the sub-goal reaches a certain threshold, called the *achievement threshold*. The achievement threshold is contained in the relation. The relation may also contain the actor of the sub-goal, if different from the actor of the obstacle.

Inhibiting: An *inhibiting relation* connects a goal to an obstacle or side-effect. The chance of the target obstacle or side-effect is set to 0 if and only if the source goal (called the inhibiting goal) is reached. In the continuous case (chance between 0 and 1), the chance is modulated, according to a parameter called the *inhibiting factor*, comprised between 0 (no modulation) and 1 (maximum modulation). Furthermore, the resulting chance also depends on the achievement value of the inhibiting goal. Symmetrically, an *exciting relation* may also be used, that will not be detailed here.

Needing: A *needing relation* connects a side-effect to a goal, different from the goal reached by the task. When the side-effect is triggered, it activates the target goal (instantiates it in the active structure), if it is not already activated. If it is already activated, it sets its achievement value to 0. In the continuous case, the target goal achievement value is diminished by a certain value, called the *decrement*. The relation may also contain the actor of the triggered goal, if different from the actor of the side-effect.

An *abstract structure* is a graph containing nodes and relations. It corresponds to what the author writes, in terms of abstract narrative content (data needed to display this content is described separately). It can be observed that such a structure does not contain any characters, and therefore does not represent any concrete narrative. For the narrative to become concrete, some goals, and their associated elements in the structure, must be *activated*. Activated elements constitute the *active structure*, which contains some elements described in the abstract structure, completed by the information about actors. The active structure is one instantiation of the abstract structure. Algorithms that build and manage the active structure from the abstract structure will not be detailed here. They are derived from [37]. In particular, these algorithms specify what changes in the target element when a relation fires.

Abstract structures do not contain variables, except the actors. In this model, it is therefore not possible to represent the fact that a character can steal an object in general. This choice is motivated by the search of a simpler model, for a first version. Variables being a powerful way of producing variations in a computer-generated narrative, they will be considered for a later version of the model.

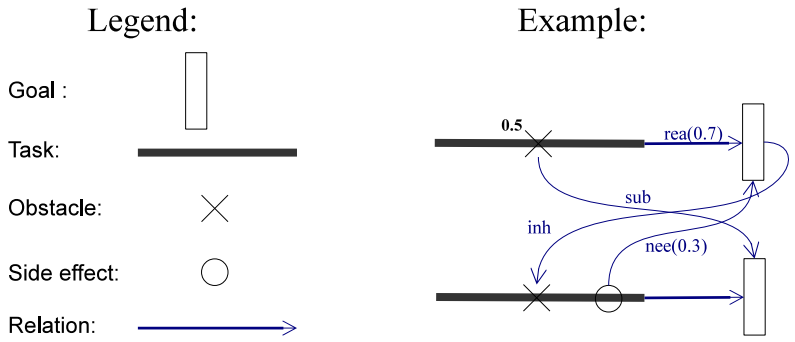

Figure 1 Graphical representation of the elements of an abstract structure. The type of relation is indicated by three letters: *rea* for reaching, *sub* for sub-goaling, *inh* for inhibiting and *nee* for needing. When the relation is associated to a value, this value is represented after the three letter code, between parenthesis. The attachment relation is not depicted, but simply represented by positioning the obstacle or side-effect on the task it is attached to. The number next to the obstacle is the associated chance. If the letter *F* is written instead, it means that this is a free obstacle (see text).

The nodes and relations can be represented visually, which is essential for the authoring process. Figure 1 depicts an example of an abstract structure containing all types of nodes and relations introduced above. Further examples are provided in Figures 2 and 5.

2.3 Adding hierarchy

When modeling a narrative at the higer level, it is important to keep the structure simple, avoiding the resulting graph to become undecipherable. At the same time, the narrative structure needs a significant amount of content to be able to express variety and richness. To solve this dilemma, one modeling technique should be added: *modularity*. It consists in grouping subsets of elements into modules, and in reasoning at the level of theses modules rather than at the level of the elements themselves. This creates a hierarchy between the lower level (simple elements, inside a module), and the upper level (modules). This approach is classical in Artificial Intelligence and has proven useful, both for computational and ergonomical reasons; see for example Hierarchical Finite State Machines [16], hierarchical petri nets [3], hierarchical neural networks [19], etc. Only two levels of hierarchy are considered in this paper.

The concept of *task module* is introduced. A task module is a subset of tasks, goals, obstacles and side-effects. This subset as a whole can connect to a goal the same way a single task can connect to a goal, via a reaching link. If an internal task with no goal attached is finished, it fires the reaching link (other tasks do not need to succeed). In the continuous case, the increment of the reaching link is specified for each internal task.

Other relations and firing mechanisms are also introduced, to and from a task module:

- Because a task module contains side-effects and obstacles, it can also connect to a goal via a needing link.
- An internal side-effect can also fire the reaching link of the enclosing task module.
- An internal side-effect or goal can be modulated by an external goal, via an inhibiting or exciting relation.

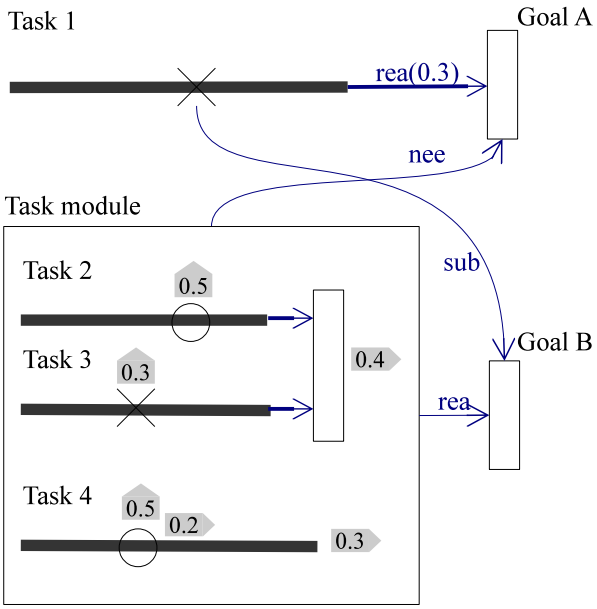

Figure 2 Graphical representation of the hierarchical and modular model for structures. Small grey arrows pointing up denote the decrement associated to the needing relation (*nee*) from the task module to the Goal *A*. Small grey arrows pointing to the right denote the achievement increment associated to the external reaching link (*rea*) from the task module to the goal *B*.

An illustration of a structure with a task module is proposed in Figure 2. In this example, the core structure is still clearly visible, while the story has gained in complexity.

2.4 Structures in tension

From the above description, an abstract structure, with its set of nodes and relations, could perfectly be created in such a way it would not present any dramatic interest. For example, the author can choose 10 tasks, connected to 10 goals, and activate the 10 goals at the beginning. As a result, the story would be a random succession of unconnected actions, which does not fit with what is usually considered as an interesting dramatic situation: in this case, the unity of action is typically lacking. So far, the model is underspecified. Therefore, the key question is: what is a good structure?

We postulated previously that a good structure is a structure that contains tension in it, and that such a structure contains some *circularity* [38]. This circularity expresses characterization of a dramatic situation as a "system of forces in internal tension"("un système de forces en tension intérieure" [32, p. 42], our translation). To further formalize and specify this circularity, we propose to think the structure in terms of paradox, as introduced in Section 1.

The structure depicted in Figure 2 contains circularity. It says that if a character wishes to reach the goal *A*, he or she meets an obstacle than leads him or her to wish to reach the goal *B*. But to reach the goal *B*, one hinders the goal *A* (needing relation from the task module to the goal *A*). In short, if this character starts to reach goal *A*, he or she does not reach goal *A*. As formulated by Nichols [28], this impossible situation creates the dynamics of narrative, when characters try to solve this paradox.

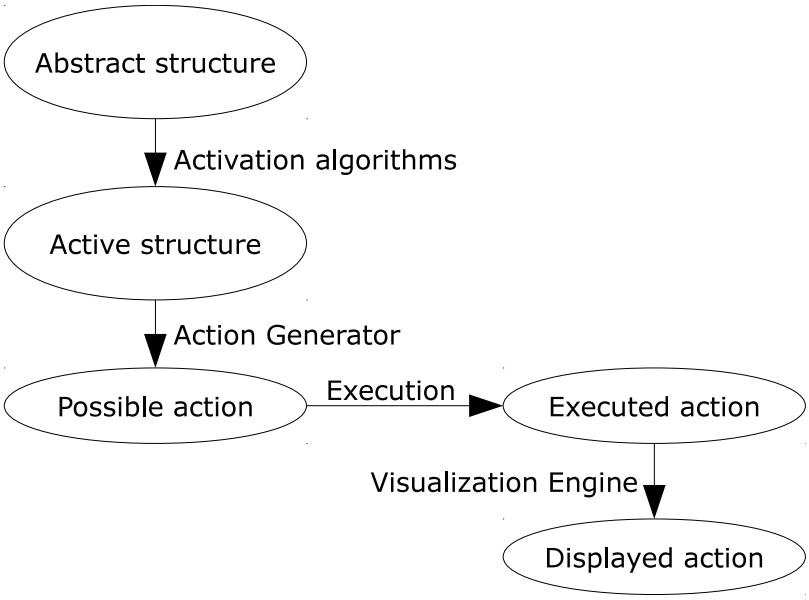

Figure 3 Causal chain of an action, from the abstract structure to its display.

Let's fill the structure depicted in Figure 2 with some narrative content. John desires that princess Mary accepts him as a future spouse (goal *A*). For that purpose, he proposes to her (task 1), but she cannot accept, because he is not a prince (obstacle). So he decides to become a fake prince (goal *B*, triggered as a sub-goal), which he can perform via different tasks, grouped into a task module. Performing these tasks of appearing like a prince makes him a liar, which in turn makes Mary not want to marry him, via the side-effect (Adapted from the animated movie *Aladdin*, Disney, 1992).

2.5 Playing around the structure

Tasks, obstacles and side-effects do not equate to what is executed and finally displayed to the user. The engine needs to calculate an action, derived from the active structure by applying a predicate on one or more of its constituting elements. The first predicate is PERFORM. It takes as parameters the task to perform and, optionally, the list of obstacles and side-effects that will trigger during the execution. An example of performance action is: PERFORM(ProposeTo(John,Mary)).

When an action is executed, it is added to the *story*. An executed action is possibly *displayed*, in which case the player can perceive it (visually, textually, auditory, etc.). To sum up, when the player perceives an action, it comes from elements in the abstract structures, which have been activated, then transformed into an action, which has been executed and finally displayed (figure 3).

Many other types of action can be found in narrative. In previous research [36], we have identified influences (encourage/dissuade), sanctions (felicitate/condemn), delegations (ask for help, propose help, accept help), etc. But reducing the types of actions to a predetermined set of possibilities proved limiting, in some contexts [33]. Therefore, we leave it open to the author which action types should be implemented, in addition to the PERFORM one. This supposes that a sort of language is designed that enables an author to program action types. For example, an author might specify the action type ASK_ADVICE, which would

generate the action: ASK_ADVICE(John,Bill,ProposeTo(John,Mary)), meaning that John asks advice to Bill concerning the idea of proposing to Mary.

These predicate-based actions are generated by author-defined rules that specify possible actions, that is actions that are logically compatible with the current active structure and the story so far. For example, a rule would specify that if a character has a goal, then he or she can ask advice to any other character, regarding any task that reaches that goal. These rules are called *possibility rules,* they are managed by the Action Generator.

Other rules must specify priorities among possible actions, so that the engine can select which actions, among all possible actions, are finally executed. For example, a rule would favor the action of asking for advice regarding a task, just after the character has been attached to the corresponding goal, in the active structure. These rules are called *preference rules*, they are managed by the Narrative Sequencer. Preference rules could be refined with a mechanism that looks ahead at the consequences of possible actions, using planning technology [43].

Finally, the visualization engine transforms the predicate-based action into a perceivable output that may include text, image, sound, three-dimensional simulation, etc. In particular, some text generation algorithms are necessary at this level.

2.6 Tension measurement and management

In the previous sections, we have provided a construction kit for structures (Sections 2.2 and 2.3), a sketch of a second construction kit for actions (Section 2.5) and a guideline for creating interesting core structures, based on the notion of paradox in narrative (Section 2.4). The next step is to enable the Narrative Sequencer to select an action according to an evaluation of the dramatic tension.

Following the view that a narrative paradox, by its logical impossibility, arouses the fundamental dynamics of narrative, we postulate that tension is created by this movement. More precisely, we propose that one important strategy to increase the dramatic tension is to force the focus of the audience to switch from one proposition in the paradox to another one. Effectively, it can be supposed that these switches activate the paradox by a phenomenon of recency: the audience lives the current position while recalling the previous one.

For example, in a two goal structure such as the one depicted in Figure 3, tension will increase if the action of mentioning the goal A to the user is followed by the mentioning of the possible failure of a task reaching this goal. Dramatic tension would be, at least in part, a matter of contrast between successive actions.

This can be represented by a "paradox space", in which each action can be positioned, depending on its relation to the success or failure of the goals (see Figure 5). By construction of the paradox, it is impossible to fully reach the goal A. But an action can still be at the right side of the paradox space – success of goal A –, because this position is temporary, or partial (achievement value is lower than one), or anticipated, or because this success is simply evoked, in a dialogue. Therefore, axes in the paradox space do not measure the achievement value of the respective goals, but a certain relation between the action and the goal, as illustrated by the following examples:

- If a character encourages another one to perform a task by mentioning the reaching of the goal, the action is placed in the positive half-plane of the space, regarding this goal.
- If a character dissuades another one to perform a task by mentioning the failure of the other goal due to a side-effect, the action is placed in the negative half-plane of the space, regarding this second goal.

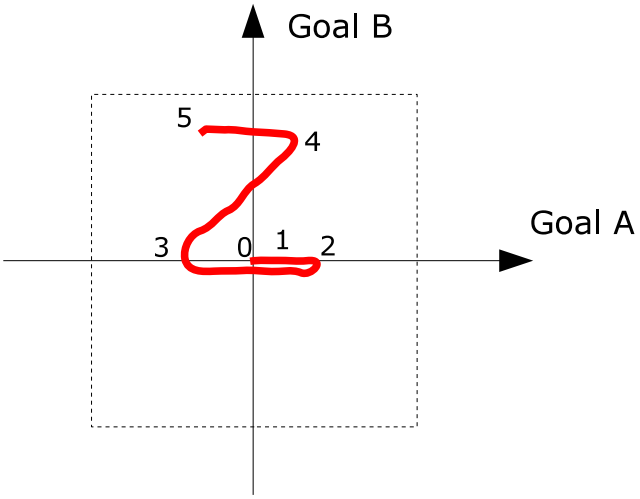

Figure 4 The paradox space, and a possible trajectory in this space (see text).

- If a character delegates a goal to another one, the action is placed in the positive half-plane of the space, regarding this goal (because it generates hope that the goal will be reached).
- Etc.

Precise estimation of the position is not described in this paper because it needs further investigation and formalization.

Let us illustrate a trajectory in the paradox space with the help of Figure 4. The trajectory is the succession of the coordinates of visualized actions in the space (only actions related to goal A or goal B are reported). Some complex actions contain several coordinates in the space. The trajectory is represented continuously for the purpose of illustration. The situation starts at the centre (point 0): the story has not started and the user's perception is neutral. The user is then considering goal A (point 1), the user attempts (point 2) to reach this goal but fails (point 3). After that, the user is faced with the possibility to solve the problem raised by the obstacle by undertaking an action that reaches the goal B (point 4), but while succeeding the task, a side-effect makes him or her fail regarding goal A (point 5), etc.

We will not give formulas of tension calculation at this stage, but only describe the main features of this calculation:

- It is initialized at 0.
- When, in the paradox space, the trajectory crosses an axis between two displayed actions, the tension is increased, proportionally to the distance between the two successive actions in the paradox space.
- The dramatic tension decays with time, meaning that if nothing happens or if executed actions do not impact the position in the paradox space, then tension decreases progressively.

In order to increase the tension, the Narrative Sequencer will adopt a strategy that consists in regularly choosing an action that crosses an axis in the paradox space.

At the same time, it is necessary to visualize the action differently, according to the level of dramatic tension the Narrative Sequencer wants to produce. This is difficult to do in a generic manner, and it highly depends on the medium. For example, if the medium allows

dynamic facial expression, then the emotions can be of higher valence when the tension is higher.

2.7 Endings

How does the story end? From a narrative perspective, for the story to end, the paradox needs to vanish. In a happy ending, something changes radically in the situation of the characters, which makes the paradox suddenly inexistent, and the main goal can be reached. For example, the law that states that a princess should necessarily marry a prince is abrogated. In a sad ending, the tension rises but the paradox remains. The tension becomes unbearable for the main character, until the main goal of the paradox disappears: the hero dies, the hero renounces, the main goal does not exist anymore (without being solved) – e.g., the princess marries another prince. In both happy and sad endings, the resolution of the paradox lies, by definition, outside of the paradox (what changed in the paradox logic? How the goal disappeared?).

Therefore, it is certainly not possible to automatically fully generate an ending, based on the terms of the paradox and general knowledge. At the general level, *types of ending* must be written by the author. Nevertheless, this does not mean that each ending is fully pre-written, because each ending can generates variations (see Section 3.4).

According to the computational model presented so far, it is possible to classify several types of endings. We will reason with the two goal case, as depicted in Figure 4. A first line of separation is naturally the sad vs. happy ending, which could be matched with the success or failure of the main goal. Another line of separation could be the success or failure of the second goal. In Figure 4, this makes four types of endings, corresponding to the four quadrants.

The role of the author is to write a certain number of endings, that she classifies in the four quadrants depicted in Figure 4. Finally, we need to specify how the narrative engine will select one of the four quadrants. The solution that we propose is in two steps:

First, each ending needs to be adaptive, so that, for example, the reason why one character changes its view should be explained by a specific action that the user has performed. More precisely, each ending is a script with some steps being activated according to a set of preconditions on the user's actions. This means that each ending has two or more variants. Furthermore, at least one ending has a variant by default, which can be chosen even if no condition on the user's actions is met.

Second, when the dramatic tension has reached a certain level (and possibly after a certain duration), an ending is triggered, among all possible endings for which one variant's preconditions are met. If several endings meet this criterion, one is chosen randomly. If no ending meets this criterion, one ending that has a default variant is chosen randomly.

3 A full example

3.1 Setting

This example is inspired by a research project that aims at helping teenagers cope when one of their parents suffers from a Traumatic Brain Injury, by offering them an interactive narrative related to their situation [18].

The user plays Frank, age 15 year, whose father, Paul, suffers from a Traumatic Brain Injury. Julia, a classmate of Frank, visits Frank at home to get advice on some math problems. When she arrives, and later in the scenario, the user has the possibility to either explain her

the family situation, or to directly help her with the math problems, which is a good thing if he wants to be appreciated by her (his goal in the story). Not explaining the situation might create problems later in the story when Paul interrupts the math session, while explaining too much might make Frank appears less cool, in front of his classmate.

From this context, the fundamental paradox is: "If I want Julia to like me, I need to help her with her math and to avoid any problem with my father I need to explain her the situation; but if I explain her the situation, then she will not like me". The structure is atemporal in that it does not say that "get math book" should occur before or after "explain accident". Of course, temporal relations can be immediately deduced from this structure, such as the execution of "explain accident" occurs before the reaching of the corresponding goal, "Julia prepared", but this corresponds to only one possible trajectory. For example, delegating the goal "Julia prepared" could occur before any attempt to perform "explain accident".

3.2 Structure

The corresponding abstract structure is depicted in Figure 6. Thanks to the two task modules, the core structure is clearly identified. The first task module contains two goals: one for helping Julia in math, and the other to deal with the father, who is otherwise constantly interrupting them. Obstacles and side-effects in the structure often fire external links, which gives them a central role in the narrative, beyond the role of providing a variant.

When instantiating the structures, characters are added: Frank (the teenager), Julia (the classmate), Paul (the father), Lili (the sister) and Olivia (the grandmother). The goal "appreciated by Julia" is instantiated, with Frank as actor. This automatically instantiates the goal "Julia ok with math" and the task "compliment Julia", to constitute the active structure. This active structure will be completed by other goals and task during the execution. Note that in this simple scenario, other characters have no goal at the beginning – they behave in reaction to Frank's actions.

3.3 Actions types and rules

In order to set in motion the active structure, action types must be defined, in addition to the default action type PERFORM. Each action type takes as parameters elements of the active structure to build a specific action. For this example, the following action types have been designed:

- Asking advice concerning a task (ASK_ADVICE): A character asks advice concerning the performance of an action.
- Asking advice concerning two tasks (ASK_ADVICE2): A character presents two alternatives to another character, in order to get advice.
- Influences regarding a task (ENCOURAGE/DISSUADE): A character encourages or dissuades another character to perform a task. Several variants are possible: influence referring to the consequence of the task (suffix: _CSQ), influence referring to a previous performance of the task (suffix: _PREV), influence referring to more than one consequences (suffix: CSQN).
- Comparing two tasks (COMPARE): In a dialog, a character compares two alternative tasks that reach the same goal.
- Help seeking (DELEGATE): A character delegates a goal to another character who will undertake the goal. It is supposed that whenever the second character has managed to

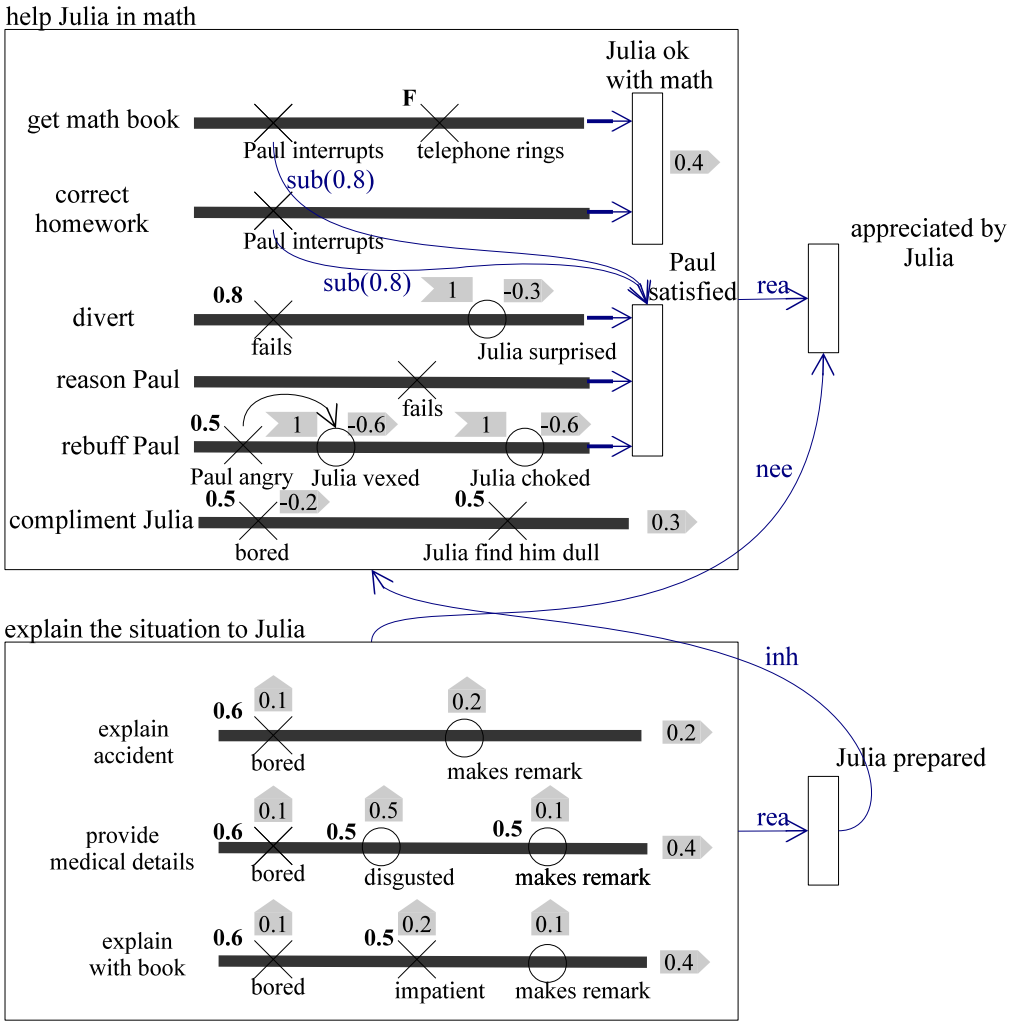

Figure 5 A full example of a narrative structure. The small arrow between "Paul angry" and "Julia vexed" means that the triggering of the side-effect is only considered if the obstacle has triggered. In the story, when Paul is angry, he becomes particularly aggressive and says bad words against Julia, so she might be vexed. But her reaction depends on wether she is prepared or not for this kind of situation (second main goal).

reach the goal, the first characters is immediately aware of it. Note that not all goals are delegable (defined by the author).
- Accept or refuse to help (ACCEPT_DLG, REFUSE_DLG): A character who has been asked for help accepts or refuses to help.
- Inform about a regret (REGRET): A character tells another character that s/he should not have performed a given task.
- Confirm regret (CONFIRM_REGRET): A character agrees with another character that the latter should not have performed a task.

For example, the action type "COMPARE" could generate the action COMPARE(Olivia, Frank, divert(Frank), reason_Paul(Frank)), which may be displayed with the following dialog line: "You could divert Paul but Julia might not appreciate your behavior. You could also try to reason Paul but he will certainly not listen to you".

Possibility rules and preference rules must also be defined, in order to transform elements in the active structure into executed actions. They have not yet been specified for this scenario.

3.4 Endings

Four families of endings can be distinguished, corresponding to the four quadrants described above, with the two following goals: "appreciated by Julia" and "Julia prepared" (respectively goals A and B in Figure 4):

Quadrant 1: Frank has pleased Julia by helping in math. For that, he has successfully explained the family situation to her. Finally, Julia finds Frank very brave and she even admires him, instead of being disgusted, vexed or annoyed.

Quadrant 2: Frank has managed to explain the family situation to Julia, but, as a consequence, Julia find him uninteresting, regardless his effort with math.

Quadrant 3: Despite Frank's efforts, Julia does not feel closer to Frank after the visit, neither has frank managed to explain her the family situation.

Quadrant 4: Frank has not managed (or even tried) to explain the family situation to Julia, because he did not want to offend her. As a result, because Paul always interrupted them, Frank has been unable to do anything to explain the math. But Julia was charmed by Frank's behavior in such a situation.

This example clearly distinguishes happy endings (quadrants 1 and 4) from sad endings (quadrants 2 and 3). It also illustrates that the happy endings need creative thinking to re-interpret the paradox and annihilate it (Julia finds Frank brave or charming). Figure 6 illustrates one of the ending, described as a script.

3.5 Output story

From the abstract structure (Section 3.2), the action types and the associated rules (section 3.3) and the story endings (Section 3.4), a story can be generated. The active structure is updated after each executed action. The following depicts a handmade outcome of this process, covering the beginning of the story, after the introduction. The text visualization has been written according to what a template-based text generation system could provide.

We first present the **formal action**, then the text-based visualisation.

Julia to Frank: "I gotta go now. Thanks for the explanation but I think I'll manage it differently next time." [event:ending2] "Bye!" Julia leaves. Olivia to Frank: "You know, it is difficult for a young girl to feel comfortable with your father". Frank leaves the room, slamming the door.
where the part in brackets refers to the description attached to an event that pushed in the Quadrant 2 during the story:

- For "explain accident": "you couldn't stop explaining your father's accident... you are becoming a drag"
- For "provide medical details": "It is so disgusting these brain operations you describe!"
- For "explain with book": "I came to study math, not medicine!" (showing the medicine book Frank showed her)

Figure 6 Script for ending 2.

1. **PERFORM(get math book(Frank)):**
   - obstacle "Paul interrupts" triggered

   Frank to Julia: "I would like to get you a great math book that will help you". He walks towards his bedroom to get the math book, but Paul comes to him and declares: "Frank, who is this stranger in our house?". Frank answers: "It's Julia Dad, my classmate, she is here to study math with me."
   Frank decides to do something to satisfy Paul's curiosity.
   Comment: *This adds the goal "Paul satisfied" into the active structure (visualized opposite as a narrator sentence)*
2. **PERFORM(divert(Frank)):**
   - obstacle "Fails" triggered
   - side-effect "Julia surprised" triggered

   Frank: "Dad, well, would you please look at the DVD player, it does not seem to work properly."
   Paul: "Oh no, I prefer staying here, with you and your friend."
   Julia: "Frank, you really have a strange way to talk to your father..."
   Frank decides to get Julia aware of the situation, so that she understands his behavior with his father.
3. **ASK_ADVICE(Frank, Olivia, explain accident(Frank))**
   Frank to Olivia: "Grandma, do you think I should shortly explain Dad's accident to Julia?"
4. **ENCOURAGE_CSQ(Olivia, Frank, provide medical details(Frank))** Olivia: "I think you should rather give her all medical details concerning your father's problem. Explain her what a traumatic brain injury is, and she has better chance to understand our family.
   Comment: *The consequence related to the encouragement is the goal "Julia prepared".*
5. **DISSUADE_CSQ(Lili,Frank, provide medical details(Frank))**
   Lili: "No no! If you give Julia all medical details, she will certainly be disgusted!" The consequence related to the dissuasion is the side-effect "disgusted".

6. **PERFORM(provide medical details(Frank)):**
   - obstacle "disgusted" triggered

   Frank to Paul: "My father had an accident. His head was hit, and after several operations in the brain, he could not fully recover. His temporal lobe is damaged. Now, my father suffers from several troubles, including loss of memory, change of mood, etc.". Julia makes a disgust expression and answers "Ah... yeah".
7. **PERFORM(correct homework(Frank)):**
   - obstacle "Paul interrupts" triggered

   Frank to Julia: "Come on Julia, let's see your homework, I would like to help you on this". Julia seems ok with this but Paul comes to her and declares: "We are just about to have dinner. I don't think we have time for you. Please leave our house!".
8. **COMPARE(Lili,Frank, divert(Frank), rebuff Paul(Frank))**
   Lili to Frank: "If you want to get rid of Dad's annoying remarks, you could distract him with something, but, if you have not explained the situation to Julia, she might not appreciate. You could also rebuff Dad, but he could become angry and say bad things to Julia.
9. **PERFORM(divert(Frank)):** obstacle "fails" triggered
   Frank to Paul: "Dad, well, would you go to the grocery store and buy something for dinner?".
   Paul: "Oh no, I prefer staying here, with you and your friend."
10. **PERFORM(rebuff Paul(Frank)):** success
    Frank to Paul: "Dad, that's enough! Could you please leave us alone?!"
    Paul: "Ok ok, sorry."
11. **PERFORM(correct homework(Frank)):** success
    Frank to Julia: "Come on Julia, let's see your homework, I would like to help you on this". Julia: "Great!". Five minutes later, Julia appears much friendly to Frank: "Thank you for your help, I appreciate it." Compared to action 7, no obstacle is triggered: Paul is "satisfied".
12. **PERFORM(get math book(Frank)):** obstacle "Paul interrupts" triggered
    Frank to Julia: "I would like to get you a great math book that will help you". He walks towards his bedroom to get the math book, but Paul comes to him and declare: "Frank, you can't leave a pretty friend alone in the living room like that!".
13. **PERFORM(rebuff Paul(Frank)):**
    - obstacle "Paul angry" triggered

    Frank to Paul: "Dad, that's enough! Could you please leave us alone?!" Paul: "What? You want to stay with this stupid friend of yours, while your father needs you?!"
    Comment: *Julia gets vexed by Paul's remark.*
14. **DELEGATE(Frank, Olivia, Julia prepared)**
    Frank to Olivia: "Grandma, I need your help. Would you please help me to explain Julia about Dad and our family?"

The above example shows a mixture of various action types. They enable to extend the expressiveness of a core narrative structure, by both producing narrative content and emphasizing the underlying paradox in the structure. The corresponding and tentative trajectory in the paradox space is represented in Figure 7.

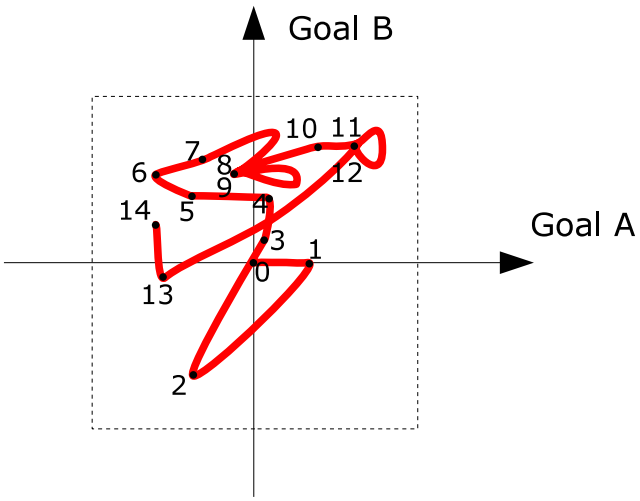

Figure 7 Tentative trajectory in the paradox space, for the story described in Table 3.5.

4 Conclusion and future work

In this paper, a novel computational model of narrative has been proposed, which comprises several components: a hierarchical model of narrative structure, a set of rules to select the next action, an algorithm to estimate the dramatic tension based on the concept of paradox, and a set of rules to select the ending. The two main contributions of this model are 1) the hierarchical nature of the model, which enables to model complex structures without loosing a global view of the main "mechanics" of the narrative (its paradox) and 2) a novel approach for modeling the dramatic tension, which renews the more usual approaches that are based on conflict or suspense.

This computational model has been only partially described above, as many components need to be more formally described: the specification of the trajectory into the paradox space, the assessment of the dramatic tension, the action selection mechanism and the endings' specification and management. In order to successfully specify these components and refine the model, the next step will consist in paper prototyping the system, according to a methodology that has proved efficient in game design [15]. In the context of interactive narrative, the model along with the proposed story will be played by the user, while the narrative management will be carried out by a "game master", who will simulate the future narrative engine. Sessions will be recorded and later analyzed.

The story described in Section 3, while going beyond the level of a "toy" example, remains relatively simple. Once the model has been validated with this story, several improvements of the model will be considered: more than one paradox in a story (e.g., one paradox per main character), more than two levels in the hierarchal structure, parametrization of narrative elements, more variations in endings.

Acknowledgments. The authors would like to thank the reviewers for the quality and the quantity of comments and advices.

References

1. Ruth S. Aylett, Sandy Louchart, João Ferreira Dias, Ana Paiva, and Marco Vala. FearNot! – an experiment in emergent narrative. In Themis Panayiotopoulos, Jonathan Gratch, Ruth Aylett, Daniel Ballin, Patrick Olivier, and Thomas Rist, editors, *Intelligent Virtual Agents, 5th International Working Conference, IVA 2005, Kos, Greece, September 12–14, 2005, Proceedings*, number 3661 in Lecture Notes in Computer Science, pages 305–316. Springer, 2005.
2. Paul Bailey. Searching for storiness: Story-generation from a reader's perspective. In M. Mateas and P. Sengers, editors, *Narrative Intelligence: Papers from the AAAI Fall Symposium*, number FS-99-01 in AAAI Technical Reports, pages 157–163. AAAI Press, 1999.
3. Daniel Balas, Cyril Brom, Adam Abonyi, and Jakub Gemrot. Hierarchical Petri nets for story plots featuring virtual humans. In Christian Darken and Michael Mateas, editors, *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*, pages 2–9. The AAAI Press, 2008.
4. Heather Barber and Daniel Kudenko. Dynamic generation of dilemma-based interactive narratives. In Jonathan Schaeffer and Michael Mateas, editors, *Proceedings of the Third Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, pages 2–7, Menlo Park, CA, 2007. AAAI Press.
5. Raphaël Baroni. Incomplétudes stratégiques du discours littéraire et tension dramatique. *Littérature*, 127:105–127, 2002.
6. Cristina Battaglino and Rossana Damiano. Emotional appraisal of moral dilemma in characters. In David Oyarzun, Federico Peinado, R. Michael Young, Ane Elizalde, and Gonzalo Méndez, editors, *Interactive Storytelling, 5th International Conference, ICIDS 2012, San Sebastián, Spain, November 12–15, 2012. Proceedings*, number 7648 in Lecture Notes in Computer Science, pages 150–161. Springer, 2012.
7. Claude Bremond. *Logique du récit*. Seuil, Paris, 1973.
8. Noel Carroll. *Beyond Aesthetics: Philosophical Essays*. Cambridge University Press, Cambridge, 2001.
9. Mario Cataldi, Rossana Damiano, Vincenzo Lombardo, and Antonio Pizzo. An agent-based annotation model for narrative media. In Wiebe van der Hoek, Lin Padgham, Vincent Conitzer, and Michael Winikoff, editors, *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, pages 2010–2012. ACM Press, 2012.
10. Marc Cavazza, Fred Charles, and Steven J. Mead. Characters in search of an author: AI-based virtual storytelling. In Olivier Balet, Gérard Subsol, and Patrice Torguet, editors, *International Conference on Virtual Storytelling (ICVS 2001)*, number 2197 in Lecture Notes in Computer Science, pages 145–154, Heidelberg, 2001. Springer.
11. Yun-Gyung Cheong and R. Michael Young. Narrative generation for suspense: Modeling and evaluation. In Ulrike Spierling and Nicolas Szilas, editors, *Interactive Storytelling, First Joint International Conference on Interactive Digital Storytelling, ICIDS 2008, Erfurt, Germany, November 26–29, 2008, Proceedings*, number 5334 in Lecture Notes in Computer Science, pages 144–155. Springer, 2008.
12. Jean E. Dumas, Nicolas Szilas, Urs Richle, and Thomas Boggini. Interactive simulations to help teenagers cope when a parent has a traumatic brain injury. *Computers in Entertainment*, 8(2), 2010.
13. Lajos Egri. *The art of dramatic writing*. Simon & Shuster, New York, 1946.
14. Syd Field. *Screenplay? The Foundations of Screenwriting*. Dell Publishing, New York, 1984.

15 Tracy Fullerton. *Game Design Workshop: A Playcentric Approach to Creating Innovative Games*. Morgan Kaufmann, 2008.

16 Patrick Gebhard, Gregor Mehlmann, and Michael Kipp. Visual SceneMaker – a tool for authoring interactive virtual characters. *Journal on Multimodal User Interfaces*, 6(1-2):3–11, 2011.

17 Algirdas Julien Greimas. *Sémantique structurale*. Presses Universitaires de France, Paris, 1966.

18 Nicolas Habonneau, Nicolas Szilas, Urs Richle, and Jean Dumas. 3D simulated interactive drama for teenagers coping with a traumatic brain injury in a parent. In David Oyarzun, Federico Peinado, R. Michael Young, Ane Elizalde, and Gonzalo Méndez, editors, *Interactive Storytelling, 5th International Conference, ICIDS 2012, San Sebastián, Spain, November 12–15, 2012. Proceedings*, number 7648 in Lecture Notes in Computer Science, pages 174–182. Springer, 2012.

19 Michael I. Jordan and Robert A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6(2):181–214, 1994.

20 Martin Klesen, Janek Szatkowski, and Niels Lehmann. The black sheep: Interactive improvisation in a 3D virtual world. In Giorgio De Michelis and Ulrich Hoppe, editors, *Building Tomorrow Today: Community, Design and Technology, I3 Annual Conference, Jönköping, 13-15 September 2000, Papers, Interactives Performances, Exhibiitons*, pages 77–80, 2000.

21 Arthur Koestler. *The Act of Creation*. Pan Books, London, 1964.

22 Yves Lavandier. *La dramaturgie*. Le clown et l'enfant, Cergy, France, 1997.

23 Claude Lévi-Strauss. *Anthropologie structurale*. Plon, Paris, 1958.

24 Elias J. MacEwan. *Freytag's Technique of the Drama: An Exposition of Dramatic Composition and Art*. Scott, Foresman and Company, Chicago, third edition, 1900.

25 Michael Mateas and Andrew Stern. Integrating plot, character and natural language processing in the interactive drama façade. In Stefan Göbel, Norbert Braun, Ulrike Spierling, Johanna Dechau, and Holger Diener, editors, *Technologies for Interactive Digital Storytelling and Entertainment, TIDSE 03 Proceedings*, number 9 in Computer Graphik Edition, pages 139–151, Darmstadt, 2003. Fraunhofer IRB.

26 Robert McKee. *Story: Substance, Structure, Style, and the Principles of Screenwriting*. Harper Collins, New York, 1997.

27 James Meehan. TALE-SPIN. In Roger C. Schank and Christopher K. Riesbeck, editors, *Inside computer understanding: Five programs plus miniatures*, pages 197–226, Hillsdale, NJ, 1981. Erlbaum.

28 Bill Nichols. *Ideology and the image*. Indiana University Press, Bloomington, IN, 1981.

29 Gerald Prince. *A Dictionary of Narratology*. University of Nebraska Press, Lincoln, NE, 1987.

30 Marie-Laure Ryan. Introduction. In Marie-Laure Ryan, editor, *Narrative Across Media*, pages 1–40, Lincoln and London, 2004. University of Nebraska Press.

31 Nikita Sgouros. Dynamic generation, management and resolution of interactive plots. *Artificial Intelligence*, 107(1):29–62, 1999.

32 E. Souriau. *Les deux cent mille situations dramatiques*. Flammarion, Paris, 1950.

33 Nicolas Szilas. Interactive drama on computer: Beyond linear narrative. In M. Mateas and P. Sengers, editors, *Narrative Intelligence: Papers from the AAAI Fall Symposium*, number FS-99-01 in AAAI Technical Reports, pages 150–156. AAAI Press, 1999.

34 Nicolas Szilas. A new approach to interactive drama: From intelligent characters to an intelligent virtual narrator. In John Laird and Michael van Lent, editors, *Proceedings of the AAAI Spring Symposium on AI and Interactive Entertainment*, number SS-01-02 in AAAI Technical Reports, 2001.

35 Nicolas Szilas. A computational model of an intelligent narrator for interactive narratives. *Applied Artificial Intelligence*, 21(8):753–801, 2007.

36 Nicolas Szilas. Requirements for computational models of interactive narrative. In Mark Alan Finlayson, editor, *Computational Models of Narrative: Papers from the 2010 AAAI Fall Symposium*, number FS-10-04 in AAAI Technical Reports, pages 62–68, Menlo Park, California, 2010. AAAI Press.

37 Nicolas Szilas, Urs Richle, and Jean E. Dumas. Structural writing, a design principle for interactive drama. In David Oyarzun, Federico Peinado, R. Michael Young, Ane Elizalde, and Gonzalo Méndez, editors, *Interactive Storytelling, 5th International Conference, ICIDS 2012, San Sebastián, Spain, November 12–15, 2012. Proceedings*, number 7648 in Lecture Notes in Computer Science, pages 72–83. Springer, 2012.

38 Nicolas Szilas, Urs Richle, and Paolo Petta. Structures for interactive narrative: An authored-centred approach. Technical Report TECFA12-1, TECFA-FPSE, University of Geneva, 2012.

39 Eugene Vale. *The technique of screenplay writing.* Grosset & Dunlap, New York, 1973.

40 Stephen G. Ware, R. Michael Young, Brent E. Harrison, and David L. Roberts. Four quantitative metrics describing narrative conflict. In David Oyarzun, Federico Peinado, R. Michael Young, Ane Elizalde, and Gonzalo Méndez, editors, *Interactive Storytelling, 5th International Conference, ICIDS 2012, San Sebastián, Spain, November 12–15, 2012. Proceedings*, number 7648 in Lecture Notes in Computer Science, pages 18–29. Springer, 2012.

41 Paul Watzlawick, Janet B. Bevelas, and Don D. Jackson. *Pragmatics of Human Communication.* W. W. Norton & Company, New York, 1967.

42 Peter Weyhrauch. *Guiding Interactive Drama.* PhD thesis, Carnegie Mellon University, 1997.

43 R. Michael Young, Mark O. Riedl, Mark Branly, Arnav Jhala, R. J. Martin, and C. J. Saretto. An architecture for integrating plan-based behavior generation with interactive game environments. *Journal of Game Development*, 1(1):51–70, 2004.