A Nondeterministic Temporal Planning Model for Generating Narratives with Continuous Change in Interactive Storytelling

Paulo Abelha, Vinicius Gottin, Angelo Ciarlini, Eric Araujo
Federal University of the State of Rio de Janeiro
Avenida Pasteur, 458, Urca. Rio de Janeiro, RJ. Brazil.
{paulo.abelha, vinicius.gottin, angelo.ciarlini, eric.araujo}@uniriotec.br

Antonio Furtado, Bruno Feijo, Fabio Silva
Pontifical Catholic University
Rua Marques de Sao Vicente, 225, Gavea. Rio de Janeiro, RJ. Brazil.
{furtado, bfeijo, faraujo}@inf.puc-rio.br

Cesar Pozzer
Federal University of Santa Maria
Avenida Roraima, Campus Universitario. Santa Maria, RS. Brazil.
pozzer@inf.ufsm.br

Abstract

In this paper, we propose a temporal planning model for real-time generation of narratives in Interactive Storytelling systems. The model takes into account continuous branched time and the specification of constraints defined as temporal formulae over dramatic properties of the narrative (e.g. *joy* or *tension*). In order to address real-time generation, dramatic properties are modeled as varying linearly and events go through a preprocessing stage. As proof of concept, the model is incorporated into an existing storytelling system, LOGTELL, which provides a way to logically specify genres; allows user interaction to influence events in the unrolling narrative; and dramatizes the story in a 3D computer graphics world. To illustrate the generation of narratives, we present a simple narrative example in the Swords and Dragons genre.

Introduction

Interactive Storytelling has arisen as a new form of media, and there are many promising systems being developed (Arinbjarnar, Barber and Kudenko 2009), where a narrative is composed and told under the influence of user interaction. There are many applications for Interactive Storytelling Systems (ISSs), such as: games (Hartsook 2011); education (Silva 2011); industry (Kline and Darken 2009); and interactive TV (Camanho *et al*. 2009). In order to provide good user experience, ISSs must generate enticing narratives, and be open to interactions while maintaining narrative coherence (Camanho *et al*. 2009).

Our contribution is to provide a model to enable the generation of narratives exploring nonlinearity and management of dramatic properties (e.g. the tension level or the overall joy) that vary continuously as the story progresses. We also consider the restriction of ISSs that work under a parallel generation-dramatization paradigm, in which the user has a continuous experience of the complete story.

Nonlinearity in ISSs (Bosser, Cavazza, and Champagnat 2010) directly relates to an important measure of quality of user experience: *replay value*, the possibility of generating many coherent and interesting stories within the same narrative genre (Silva 2010). Management of dramatic properties throughout the story can be used to enhance user experience (Zagalo, Barker, and Branco, 2004) and conform narratives to specific aesthetic requirements, such as desirable tension arcs (Barros and Musse 2008) and (Riedl and Young 2004).

We address nonlinearity through the adoption of nondeterministic events, with more than one way of reaching each ending. Events are combined to form a nonlinear narrative through a two-stage planning approach: first we use a partial-order planner to generate a sketch and, second, we detail it with a planner that applies HTN tech-

niques (*Hierarchical Task Networks*) (Erol, Hendler, and Nau 1994) and acknowledges the verification of temporal restrictions dramatic properties (e.g. *tension*, *joy*, *anticipation* etc.). These arcs are modeled by continuous values that vary in a continuous branched time as the nonlinear story progresses.

We acknowledge continuous values as a way of capturing aesthetic facets of storytelling more accurately than discrete values. The expression of the variation of these values in time is achieved through the use of a temporal logic that is an extension of CTL (*Computation Tree Logic*) (Konur 2012) and was first proposed by (Araujo *et al*. 2011). With this logic, we define typical temporal restriction formulae for a genre, such as "*there is always a happy ending*" for fairytales; or "*the tension level may not go below a certain threshold*" in a suspense setting. The logic allows for more complex and conditional expressions, like "*there is always the possibility that, if the story becomes really sad, it will become joyful again*".

The approach described in (Araujo *et al*. 2011) considers only the verification over pre-generated plots, in a generate-and-test fashion. That is, first it generates a chapter of the story, and then, verifies if it satisfies the temporal formula. However, such an approach is certain to face difficulties when handling a real-time story generation paradigm, with a large number of events and possible outcomes. When verifying temporal restrictions over chapters composed of nondeterministic events, the number of possible combinations – i.e., possible *chapters* – becomes exponential. Therefore, a generate-and-test approach will not obtain a valid chapter in non-optimal cases.

In order to avoid the overhead of generating an entire chapter before verifying that it is invalid, we incorporate the verification of temporal restrictions in the plot generation process. Still, our time restrictions are strict, for we consider a simultaneous dramatization and generation process approach, where a user experiences a chapter while the system concurrently generates possible subsequent chapters. In addition to eliminating the generation-and-test-approach, we apply a previous, *offline*, method for the verification of temporal formulae to generate an indexed data structure that holds information of the conditional satisfability of temporal formulae of every event and possible outcome.

Content Model

We assume a content model based on a formalism that specifies Interactive Storytelling *genres* (Ciarlini *et al*. 2010). Here, we mention only what is necessary for a complete understanding of our plot generation process. The formalism is currently used in LOGTELL (Ciarlini *et al*. 2010), which is a storytelling system capable of generating and dramatizing interactive stories in a 3D animation environment while also handling user interaction.

We define a genre (e.g. *Swords & Dragons*) as containing the necessary data for generating all plots sharing common characteristics: character archetypes; typical story developments; and a fixed repertoire of events. Additionally, we define typical temporal restrictions over dramatic properties.

Character archetypes define all *entity classes* and their associated *attributes* – e.g., the *Knight* and *Dragon* classes, both possessing the *strength* and *evilness* attributes. Typical story developments are defined as a set of *goal inference rules*, specified in a modal temporal logic (a different logic than the one we use for restrictions over dramatic properties), capturing and expressing both entities' conditional motivations and common narrative goals. For example, an evil *Knight* or *Dragon* will try to kidnap any *Princess* if she is not protected; and a good *Knight* will try to heroically save an endangered *Princess*.

An event is an occurrence in the story world, and is represented in the planning process as a parameterized operator, with preconditions and effects over context properties (entities and their attributes). For the HTN planner, events correspond to tasks in HTN, as explained in (Silva 2010). *Primitive* tasks correspond to ***basic events*** (e.g. *Go*) and *nonprimitive* tasks correspond to ***complex*** events, which can be ***composite***, ***abstract*** or both. All operators have preconditions, but only basic operators have effects.

Composite events encapsulate sequences of other events, and correspond, in HTN, to tasks with subtasks (e.g. "Villainy", composed by the sequence "Go", "Assault", "Victimize" and "Go"). Abstract events represent generalizations that can be alternatively specialized into other events (approaching the concept of *method* in HTN). Additionally, events can be both abstract and composite so that they can be derived into distinct sequences of events (e.g. "Heroism", which may be specialized either into "Go" and "Rescue" or into "Go" and "Vengeance").

Basic events are the only ones that can be dramatized and, as such, a nondeterministic automaton containing dramatization information is associated with each one of them. Therefore, each path on the automaton represents a possible dramatization sequence of that event; and *only basic events are dramatized*, since complex events are not associated with automata. Transitions in the automaton are composed as sequences of micro-actions – which are parameterized atomic operations, directly dramatizable by the dramatization module. Besides providing information for the dramatization process, an event's automaton also defines the variation of dramatic properties within that event. Dramatic properties variation is defined in each state of the automaton as a linear combination of the *initial value for that dramatic property* – the value in the initial state.

We limit variation to linear expressions in order to avoid extreme computational costs in the verification algorithms, which use constraint solving techniques that do not handle higher-order expressions in reasonable time (Araujo *et al*. 2011). Given the typical restrictions over dramatic properties, it is possible to generate a formula satisfaction index table, containing preprocessed information about each event in the genre.

Generating Narratives

We compose narratives based on a *context*: a set of logical specifications for an initial situation, regarding entities (e.g. characters, places etc.) and values for their attributes. A context can thus be seen as a particular instantiation of a genre, defining an initial world state for plots of that genre.

We define a *plot* to be a sequence of one or more chapters. A *chapter* is a nonempty tree, where each node is an event. If an event has only one possible ending, then it does not generate branching in the tree; every event has at least one ending.

We want to generate rich narratives and for this we have specified two requisites: nonlinearity and satisfaction of dramatic properties' variation. The first one is achieved by incorporating nondeterminism into the planning process. The second one is ensured by allowing the specification of typical temporal restriction formulae that we might want to be valid in the chapters under certain conditions and preprocessing all the possible events in accordance with these formulae to build a genre's index table.

Narratives are generated in chapters. When generating a chapter, the model considers a temporal formula containing all constraints that should be satisfied by the chapter. While a plan (chapter) is being dramatized, the next possible chapters (one for each possible outcome of the chapter) are being planned.

The complete process of generating narrative chapters consists of obtaining inferred goals from the set of goal inference rules; generating a sketch plan using partial-order planning to achieve inferred goals; and detailing the sketch plan using a nondeterministic hierarchical task network planning algorithm that verifies the temporal formulae with the help of the previously built index table. Temporal formulae are only taken into account in the HTN planning phase. The final result is a tree of events. Our two-stage planning approach provides us with the efficiency of HTN planning, while maintaining the characteristic of generating plans that do not strictly follow pre-established patterns.

Genre Specific Temporal Formulae

As previously stated, every genre, as part of its specification, has a set of typical temporal formulae that describe typical temporal restrictions over dramatic properties. For instance, we could say that our *Sword & Dragons* genre is typically tense and subject to cheerless happenings. Then, one possible formula would be: "*in every storyline, the tension level is always high and there is also the possibility of the story getting sad*". We define this, and similar formulae, in a modal temporal logic that is an extension of CTL.

The logic adopted (Araujo *et al*. 2011) uses the path quantifiers **A** (all paths) and **E** (there exists as path) and the state operators **G** (always in the future)**, F** (eventually in the future) and **U** (until). The main difference from CTL is that it considers continuous time. As we have an infinite number of states, there is no way to use the state operator **X** (next state) since it is not possible to talk about a next state. Furthermore, the logic adds to CTL, the operator **end** that considers formulae validity at the end of a chapter. This operator is useful in a storytelling context because there is a typical need to restrict how a story ends (for instance, we may want force a happy ending).

We define a well formed formula by the following grammar (where *a* and b represent linear combinations of continuous dramatic properties that have specific values in each moment of a story and α and ß represent modal temporal formulae):

$$\alpha, \beta \rightarrow basic\_constr \mid AF\,\alpha \mid AG\,\alpha \mid EF\,\alpha \mid EG\,\alpha \mid A\,\alpha U\,\beta \mid E\,\alpha U\beta \mid$$
$$A\,end\,\alpha \mid E\,end\,\alpha \mid \alpha\ AND\ \beta \mid \alpha\ OR\ \beta \mid \alpha\ IMPLIES\ \beta \mid \neg\alpha$$
$$basic\_constr \rightarrow a = b \mid a \neq b \mid a > b \mid a < b \mid a \geq b \mid a \leq b$$

Semantic validation for basic constraints and formulae with logic connectives (AND, OR,¬ and IMPLIES) is as usual. Semantic validation of quantified formulae is the same as CTL, adding the notion of a continuous time and our new operator **end**. Figure 1 shows a graphic semantic representation of possible operator combinations.

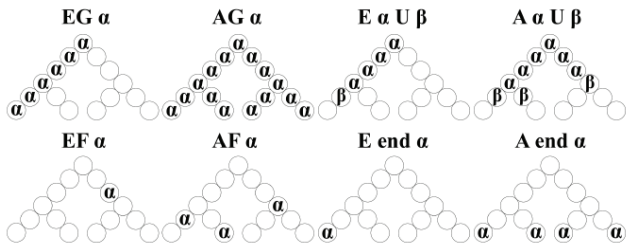

Figure 1. Graphic semantic representation of possible extended CTL operator combinations.

The aforementioned typical temporal formula ("*in every storyline, the tension level is always high and there is also the possibility of the story getting sad*") could then be defined as: *AG {tension > 60} and EF {joy < 20}*.

Preprocessing Typical Temporal Formulae

Given that our model predefines the information of a genre's typical requirements over dramatic properties, our strategy is to perform a thorough *offline* verification of satisfiability, by each event, of temporal formulae expressing these restrictions. The preprocessing step generates an indexed data structure that contains the results of this verification. The planning process queries the structure and retrieves this information in order to avoid considering the continuous variation within events. Each entry in the structure states that a temporal formula α holds in event *E* iff formula β – a *residual formula* – holds from then on. Unconditional satisfaction generates *true* as a residual formula. Likewise, *false* as a residue means the formula does not hold. Any other residual formula must be incorporated as a current requirement by the planning process.

This data structure is generated using a constraint logic programming algorithm, performing a symbolic execution of the automaton describing event's dramatic properties' variation. Since this is an abstract verification – the values of dramatic properties are not known – collected constraints are inserted as *conditions* for residual formulae to be valid. The planning process tests these conditions against the actual values of dramatic properties, and assumes all valid residual formula as suitable for continuation.

In the scope of this work, the data structure is assumed to contain the verification results of every possible formula necessary by the planning process. We provide, in the last section of this paper, an example query that describes the structure in further detail.

HTN Planning

HTN planning is an artificial intelligence planning technique that, by using context information, greatly improves efficiency (Erol, Hendler, and Nau 1994). A plan is defined as the satisfaction of a task network (a partial order of tasks), where each task can be either *primitive* or *nonprimitive*. A typical HTN planner receives a task network to be solved and recursively decomposes every nonprimitive task, by means of *methods* that specify conditions and ways of decomposing nonprimitive tasks, until it ends up with an HTN composed of only primitive tasks.

This planning technique is usually applied when there are time restrictions and available domain knowledge to design tasks and possible methods of decomposing them. This technique is very useful in our case, with the approach of a simultaneous process where chapters are generated and dramatized in parallel under a paradigm of Interactive TV, where response time is a critical factor (Camanho *et al*. 2009).

When planning a chapter, we are trying to combine events. Complex events are used solely for planning purposes, and are decomposed and specialized to become combinations of basic events. Therefore, after the planning stage is complete, we have a chapter of basic events to be dramatized.

Our HTN planner receives a sketch plan (as a partially-ordered HTN) to be detailed so as to end up with a contingency tree of basic events. If all events were deterministic, our HTN planner would generate chapters corresponding to a sequence of basic events. As we might have nondeterministic events, the planner generates a contingency tree of events, in which, after each event, the edges correspond to conditions that are tested at execution (dramatization) time in order to choose the next event. In this tree, each path from the root to an outcome of a leaf corresponds to a different version of the same chapter.

When LOGTELL generates a sketch plan, it takes into account the idea of incorporating attempts to achieve goals, i.e. some events are incorporated with annotations specifying they should be incorporated to the plan only if their preconditions are valid at the current state. Our model details the sketch, taking into consideration all possible different effects of each event and the annotations corresponding to events that should be conditionally incorporated.

An abstract event is handled by replacing it with one of its applicable specializations to form a new HTN to be planned. A composite event is replaced by its sub-HTN in the original HTN to form a new HTN to be planned. A complex event that is both abstract and composite leads its chosen applicable specialization to inherit the complex event's sub-HTN, and, additionally, the child event may add its new particular partial order restrictions to the inherited sub-HTN.

Basic events are handled by iterating over all its active endings, which are the ones that are reachable concerning the current state of the world. In the case of deterministic events, the singular ending is always reachable. For each reachable ending, we obtain the residual formula; update the value of the dramatic properties; and update the state of the world according to the set of effects for this ending. From the second reachable ending on, existentially quantified formula (e.q.f.) that were satisfied by previous sibling's subplans can be removed from the entry temporal formula to result in a possibly simplified residual formula for the current and next siblings. The last step is to remove the basic event from the HTN and continuing the planning process regarding these new values.

Temporal formulae are maintained in a normal form, where implications are removed and negations are pushed inwards to basic formulae. Regarding a normalized temporal formula, we call each of its subformulae with only one outermost quantifier, an *element* of the temporal formula.

The residual formula procedure receives as entry an event, an ending and a temporal formula, obtaining a structure containing the conditional temporal result of executing the event through a path that leads to ending regarding the current state of the world and the current state of the dramatic properties. In the case of a final event (a leaf in the plan), this procedure will consider that dramatic properties hold their current value indefinitely. This conditional temporal result is composed of a list of temporally conditioned pairs (explained below) and the residual formula itself.

The process to generate a conditional temporal result consists of running through every element α_e of the entry formula α_i, and, for each α_r as the residual formula for α_e:

- If $\alpha_r \neq true$ **and** $\alpha_r \neq$ false, add a pair (α_e,α_r) to the list of temporally conditioned pairs;
- If α_e is an e.q.f. **and** $\alpha_r = false$, remove α_e from α_i;
- If α_e is a universally quantified formula **or** $\alpha_r = true$, substitute α_e for α_r in α_i;
- Simplify α_i. If $\alpha_i = true$ **or** $\alpha_i = false$, the process stops.
- If α_e is an e.q.f. **and** $\alpha_r = true$, α_e is added to SEF (the set of satisfied e.q.f.).

If the above process results in a residual formula *false*, the planning procedure fails and backtracks to the last viable option. If the residual formula is *true*, the planner continues normally without any temporal restriction thenceforth.

When dealing with basic events, the planner iterates over every active ending, and eliminates from the entry temporal formula, existential formulae that were satisfied by previous subplans (not applicable when planning for the first ending or for deterministic events); obtains the new current state for the dramatic properties; recursively plans for the current ending; when it returns from the recursion, obtains the satisfied formulae for the current ending; verifies if the entry temporal formula is still satisfied (every universally quantified element with an existentially quantified residual formula must have this residual formula belonging to the set of satisfied formulae returned by the subplan); adds the satisfied formulae from its subplan to the total set of already satisfied formulae by every ending. At the end of the planning process, the planner verifies if every e.q.f. has been satisfied by the plan.

A Concrete Example

In order to clarify and test our model, we have created a context in the Swords & Dragons genre. This context enables the generation of a few possible chapters. The main narrative consists of a hero who goes in a quest to fight the villain after the villain has attempted to kidnap the princess. We will consider the aforementioned typical temporal formula. The dramatic properties' initial value is *70.0* for *tension* and *50.0* for *joy*.

In Figure 2, we see the sketch of our example narrative. In this sketch, we have a composite event "Villainy" in which a villain practices an act of evil, such as trying to kidnap a

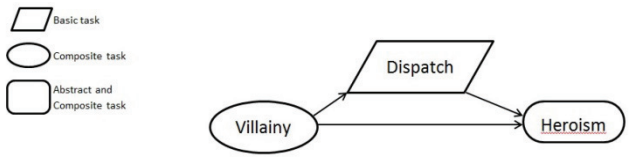

Figure 2. A sketch chapter as an HTN.

victim. There is also the event "Dispatch", in which a patron arranges a quest for a hero in order to help him to combat the villain. Finally, we have the event "Heroism", in which a hero goes to the villain's lair to complete his quest. "Villainy" and "Heroism" are events that can unfold in different ways, since the former is composite and the latter is both composite and abstract. In Figure 3, we see a total decomposition of the sketch plan. In other words, we see a snapshot showing all possible choices of specializations and decomposition that could derive from the sketch plan.

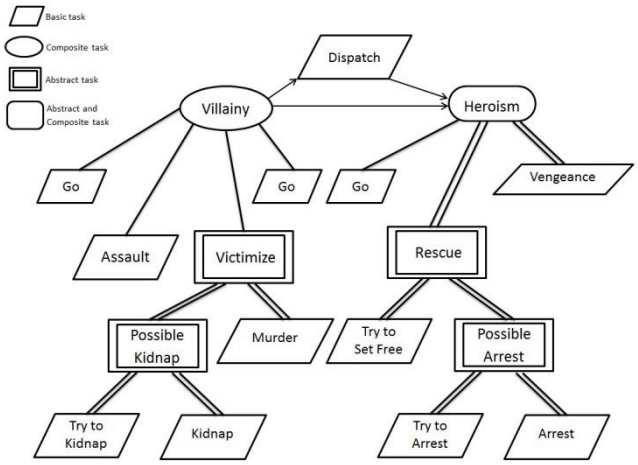

Figure 3. Total decomposition of the initial sketch. Simple edges represent a relation of component of a composite task. Double edges represent specializations of abstract tasks. Arrows represent partial order restrictions.

The HTN planner will select an event with no predecessor. In this case, "Villainy", and, since it is a composite event, it will be decomposed in its subevents: "Go", "Assault", "Victimize" and "Go". This new task network is merged with the initial one and the planner goes on to select another event with no predecessor. In this case, it is the task "Go", which is a basic event. Therefore, we need to consult for the residual formula. The event "Go" is deterministic and does not change our current (initial) temporal formula. The HTN planner proceeds to the task "Assault" which is also deterministic, with only one path, and also gives us a residual formula identical to the entry formula and, in this example, does not change the value of the dramatic properties.

When the planner gets to the "Try to Kidnap" event, it finds its first basic event for which the consult procedure returns a different residual formula than the entry formula. For each ending, we need to consult for the entry formula regarding both of its elements: *AG {tension > 60}*, and *EF {joy < 20}*. For the first ending, s_2, the consult for the first element, returns *AG {tension > 60}*. The consult for the second element returns *true* because the entry element *EF {joy < 20}* is satisfied for s_3, which implies that it is also satisfied for s_2. Since the entry formula is a conjunction, the resulting residual formula for this event through s_2 is *AG {tension > 60}*. For the second ending, the residual formula is the same for both elements, resulting in the same residual formula.

So far we have seen an example of how our planner deals with different types of events and consultations. Planning continues analogously for all other events and after it finishes, we end up with a chapter to be dramatized.

Conclusion

In this paper, we presented a model for generating narratives for interactive storytelling. We aim at enhancing user satisfaction through greater replay value and satisfaction of restrictions over dramatic properties, such as the *tension* level or the overall *joy*, throughout the story. Our approach is novel in that it considers both temporal formulae satisfaction and nonlinearity in narratives, and operates based on formal genre definition.

Narratives are generated chapter by chapter, with each one being a tree of (possibly) nondeterministic events. Generating a chapter by simple combination and verification demands an exponential amount of time on the number of possible outcomes of events (paths and endings). Such a generate-and-test approach would fail to generate complex narratives, thence our planning process assimilating the verification process in one unique approach that combines nondeterministic HTN planning with a preprocessing strategy for verification of temporal formulae over dramatic properties defined in an extension of CTL.

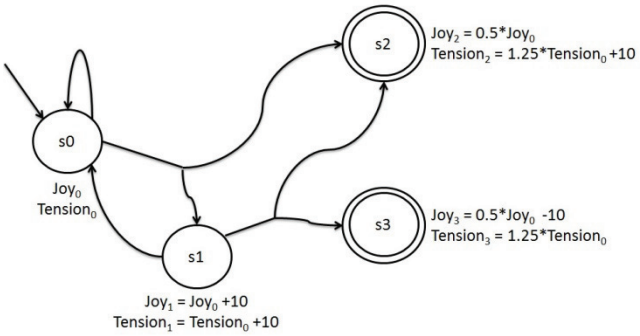

Figure 4. The automaton for the "Try To Kidnap" event. The value of dramatic properties in each state is a linear combination of their values in the initial state. The initial state is s_0 and there are two final states: s_2 and s_3.

References

Arinbjarnar, M., Barber, H. and Kudenko, D. 2009. A Critical Review of Interactive Drama Systems. In Proceedings of the AISB'09 Symposium: AI & Games, 15-26. Edinburgh, Scotland.: The Society for the Study of Artificial Intelligence and the Simulation of Behaviour.

Hartsook, K., Zook, A., Das, S., Riedl, M.O. 2011. Toward Supporting Stories with Procedurally Generated Game Worlds. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 297-304. Seoul, South Korea.: IEEE.

Silva, M.A.R., Lima, D.C., Silva, J.C.A., Souza, D.G., Martinez, C.M.S. 2011. A Narrative Game as an Educational Entertainment Resource to Teach Words to Children with Learning Deficits: A Feasibility Study. In *Proceedings of the 10th International Conference on Entertainment Computing*, 222-227, Vancouver, Canada.: Springer.

Kline, D., Darken, C. 2009. Bringing Interactive Storytelling to Industry: Designing a Reactive Narrative Encounter System. In *Proceedings of the Fifth AAAI conference Artificial Intelligence and Interactive Digital Entertainment*, Palo Alto, CA.: AAAI Press.

Camanho, M.M., Ciarlini, A.E.M., Furtado, A.L., Feijó, B., Pozzer, C.T. A model for Interactive TV Storytelling. 2009. In *Proceedings of the 7th Brazilian Symposium on Games and Digital Entertainment*, 197-206, Rio de Janeiro, Brazil.: IEEE.

Bosser, A., Cavazza, M., Champagnat, R. 2010. Linear Logic for Non-Linear Storytelling. In *Proceedings of the 19th European Conference on Artificial Intelligence*, 713-718, Lisbon, Portugal.: IOS Press.

Zagalo, N., Barker, A., Branco, V. 2004. Story reaction structures to emotion detection. In *Proceedings of the 1st ACM workshop on Story representation, mechanism and context*, 33-38, New York, NY.: ACM.

Barros, L.M., Musse, S.R. 2008. Towards consistency in interactive storytelling: Tension arcs and dead-ends. In *Computers in Entertainment* 6 (3): 43:1-43:17.

Riedl, M.O., Young, R.M. 2004. An Intent-Driven Planner for Multi-Agent Story Generation. In Proceedings of the third International Joint Conference on Autonomous Agents and Multiagent Systems, 186-193. New York, NY.: IEEE Computer Society.

Erol, K., Hendler, J., Nau, D. S. 1994. UMCP: A Sound and Complete Procedure for Hierachical Task-Network Planning. In Proceedings of the International Conference on AI Planning Systems, 249-254, Chicago, IL.: AAAI Press.

Konur, S. Real-time and probabilistic temporal logics: An overview. Retrieved May 5, 2013 from Cornell University Library: http://arxiv.org/abs/1005.3200v3.

Araujo, E.T., Ciarlini, A.E.M., Pozzer, C.T., Feijó, B. 2011. A Method to Check the Satisfaction of Continuous-Time Constraints by Nonlinear Stories. In Proceedings of the 4th International Conference on Interactive Digital Storytelling, 272-277, Vancouver, Canada.: Springer.

Silva, F.A.G., Ciarlini, A.E.M., Siqueira, S.W.M. 2010. Nondeterministic Planning for Generating Interactive Plots. In Proceedings of the 12th Ibero-American Conference on AI, 133-143, Bahía Blanca, Argentina.: Springer.

Ciarlini, A.E.M., Casanova, M.A., Furtado, A.L., Veloso, P.A. 2010. Modeling interactive storytelling genres as application domains. Journal of Intelligent Information Systems 35 (3): 347-381.: Kluwer Academic Publishers.