From Linear Story Generation to Branching Story Graphs

Mark O. Riedl
University of Southern California

R. Michael Young
North Carolina State University

Narrative as entertainment in the form of oral, written, or visual stories plays a central role in our social and leisure lives. In education and training contexts, narrative helps to motivate and illustrate. *Narrative intelligence* refers to the ability—human or computer—to organize experience into narrative. A computer system that takes a narrative approach to entertainment, education, or training can use its ability to reason about narrative intelligence to structure narrative in ways that help the user understand content.

Recently, researchers have applied narrative intelligence to create interactive narrative systems, virtual worlds in which a story unfolds and the user is considered a character in the story, able to interact with elements and other characters in the virtual world. The standard approach to incorporating storytelling into a computer system is to script a story at design time. However, this approach limits the computer system's ability to adapt to the user's preferences and abilities. Furthermore, if stories are scripted at design time, a system can only have a limited number of stories to present to the user because there are a limited number of points at which the user's actions or decisions can have a discernible impact on the rest of the story. In entertainment applications such as computer games, a limited number of stories or permutations of a single story limits the game's replay value. In educational and training applications, a limited number of stories or permutations of a single story prevents a system from fully catering to students' needs and abilities.

The alternative approach is to generate stories dynamically or on a per-session basis (one story per time the system is engaged). Narrative generation is a process that involves the selection, ordering, and presentation through discourse of narrative content (the events that will be presented to an audience). A system that can generate stories can adapt narrative to the user's preferences and abilities, has expanded replay value, and can interact with users in ways that system designers didn't initially envision.

Interactivity and narrative

Computer games and educational and training applications use two fundamental narratives types: linear and branching.

Linear narrative is a traditional narrative form in which a sequence of events is narrated from beginning to ending without variation or possibility of a user altering how the story unfolds or ends. Linear narrative is found in traditional storytelling media such as novels and movies. Computer games often use linear plots (that is, an outline of the narrative's most significant occurrences) although the story structure is partitioned into interactive portions—levels—and cut scenes. Although the user has a certain degree of control during level play, the only outcome is successful completion of some objective (usually killing all of the enemies in an area) or failure, in which case the user must try again. Therefore, you wouldn't typically consider the events occurring during level play as part of the plot although they are part of an emerging narrative. All users experience the same plot during successive sessions.

Narrative mediation, a generative approach to interactive narrative construction, can express the same stories as systems using branching narrative structures, and so can use linear narrative generation techniques.

In branching narrative, many points exist in the story at which a user action or decision alters the way a narrative unfolds or ends. Branching narratives (see, for example, Gordon et al.[1]) are typically represented as directed graphs in which each node represents a linear, scripted scene followed by a decision point. Arcs between nodes represent decisions the user can make. Although a branching narrative can introduce variability into the user's experience with a storytelling system, that variability is typically built into the system at design time and is thus limited by the system designer's anticipation of the user's needs or preferences. Users are constrained to the structure of the branching story graph such that if they make the same choices at each decision point, they'll have identical experiences with the system. That is, if a user made the same decisions during two consecutive sessions with the system, the experience would be the same.

Control versus coherence

Interactive narrative systems must balance a story's coherence against the amount of control afforded the user.[2] A narrative's coherence helps determine its understandability—that is, the user's ability to comprehend the relationships between the events in the story, both within the story world (for example, the causal or temporal relations between actions) and in the story's telling (for example, the selection of camera sequences used to convey the action to the user). Systems that construct stories should respect the user's sense of coherence by clearly linking each action in the story world to the story's overall structure. The degree of user engagement within an interactive narrative lies, to a great extent, with the user's perceived control over the character. The greater the users' sense of control over the character, the greater the sense of presence—that is, users perceive that they're part of the story world and free to pursue their own goals and desires.

Unfortunately, control and coherence often conflict in interactive narrative systems. To present a coherent narrative, the actions within a story are carefully structured (either by human designers at design time or by narrative generation systems at run time) so that actions at one point in the story lead clearly to state changes necessitated by actions occurring at subsequent points. When users exercise a high degree of control within the environment, their actions will likely change the world's state in ways that can interfere with the causal dependencies between actions intended within a story line.

Unfortunately, control and coherence often conflict in interactive narrative systems.

One reason that control and coherence conflict is that the combinatorial complexity of authoring branching stories is such that story graphs scripted at design time have either a limited number of decision points or a low branching factor (the number of alternatives in any given decision point). As the number of decision points grows, the amount of story content that must be authored grows exponentially. Not only must human designers produce a large amount of story content, some of which might go unseen by the user, but they must also author the content such that users have a coherent experience no matter what decisions the users might make along the way.

One way to overcome the conflict between control and coherence is to take a generative approach to authoring branching narrative content. A computer system that can generate a branching narrative can compute all possible branches with an arbitrary branching factor without succumbing to fatigue or losing track of details that make each possible narrative experience coherent. More decision points and higher branching factors correlate directly with the user's perception of control. But how do you generate a branching narrative? One way is to generate a linear narrative and then recursively apply a process that determines all possible decision points and generates alternative linear narratives for every possible decision.

Narrative mediation,[2,3] as used in the Mimesis interactive storytelling system, balances the tension between control and coherence in interactive storytelling by making linear narratives interactive. Assuming that the branching narrative is a model for representing any interactive story, we must ask whether the narrative mediation's expressive power is at least as powerful as the story graph representation. If it is, we can apply linear narrative generation to branching narrative generation. Our proofs show that narrative mediation is at least as expressive as acyclic branching stories and also provides concurrency of user and character action, a feature that branching stories don't typically consider.

Generating interactive narratives

Two main approaches to generating interactive narratives exist.

Autonomous agents and drama management

One technique for generating interactive narrative is to implement the system-controlled story-world characters as autonomous agents that can react to the user and the environment in a believable manner. The story emerges from the agents' decisions and behaviors in the virtual world.[4] However, the approach offers no explicit plot representation or defined notion of the story's outcome, so it's possible that no coherent narrative structure will emerge. For example, Cavazza, Charles, and Mead describe a system in which an autonomous agent controls each story world character.[5] Each agent uses an instance of a hierarchical task net planner to achieve its goals. Interesting narratives emerge when several agents' plans conflict, causing one or more agents to fail and replan.

To ensure narrative coherence in a system in which characters are implemented as autonomous agents, some interactive narrative systems use drama managers (for example, Bates,[6] Weyhrauch,[7] and Magerko et al.[8]). A drama manager monitors the activities of the autonomous characters and the user character relative to a plot graph—a partially ordered graph of events that move the story forward. When more than one event is possible, the drama manager analyzes the tradeoff of one or the other occurring next and subtly manipulates the world and autonomous characters to allow the event leading to the more satisfying experience. A plot graph isn't a representation of a branching story graph, although you can use adversarial-like search to find complete linear narratives in the space of possible narratives described by a plot graph.[7,9] Searching the complete narrative space is, unfortunately, intractable.[9]

The *Façade* interactive drama system[10] uses a drama manager that exerts strong story control over the characters. The drama manager continuously and reactively selects from a pool of miniscenes called *beats*. The drama manager selects beats on the basis of applicability constraints (for example, whether the beat is relevant to world conditions) and on the degree to which the beats fit an idealized dramatic arc. The main characters enact the behaviors encoded in the beats to create dramatic effect and move the story line forward. The story line plays out coherently and a user perceives that his or her choices affect the story's direction. However, the *Façade* story line's coherence is partially due to content authors' ability to provide a sufficient pool of beats to cover all alternative plot sequences. Authors must write every possible beat's content ahead of time, although the system determines the content's ultimate sequencing intelligently and dynamically at run time.

Narrative mediation

Another technique for generating interactive narrative is narrative mediation,[2,3] which gives a centralized author agent control of character actions. The system generates a linear narrative representing the ideal story to tell the user and then considers all the ways that the interactive user can interact with the world and with the other characters. The generated story includes actions that system-controlled characters perform as well as actions that the user-controlled character should perform. For every action the user makes that threatens to deviate too severely from the system's proposed linear story, the system dynamically generates an alternative story line from the deviation point.

Narrative planning. In narrative mediation, a plan represents the story. The plan contains annotations that explicitly mark the temporal relationships between all actions (by both user- and system-controlled characters) in the plan, indicating the steps' execution order. Planners use other annotations, called *causal links*, to mark all causal relationships between the actions in the plan. A causal link connects two plan steps s_1 and s_2 via condition e (that is, $s_1 \rightarrow^e s_2$) when s_1 establishes the condition e that subsequent action s_2 needs to execute.[11] As users issue commands for their character to perform actions in the story world, the system checks the actions against the story plan to determine whether they're exceptions. Exceptional actions threaten conditions in the world required by future system-controlled character actions. Specifically, an exception occurs whenever a user attempts some action α, where some effect $\neg e$ of α threatens to undo some causal link $s_1 \rightarrow^e s_2$ where s_1 occurred prior to α and s_2 has yet to occur.

Causal dependency planning[11] operates in a backward chaining fashion as a flaw-repair process. A flaw is an annotation on an incomplete plan that specifies how the plan will fail to execute. The planning process revises a flawed plan into child plans, with each sibling representing a different way to repair the same flaw. The child plans themselves have flaws inherited from the parent plan or introduced when other flaws are repaired. Planning is thus a search for an unflawed plan through a tree of partial plans. In this approach, the planner is initialized with a root plan—typically an empty plan containing goal propositions that must be made true in the world. The goal propositions are open conditions in the root plan (that is, flaws indicating that conditions aren't yet marked as established). To repair open conditions, the planner extends a causal link from a preceding step in the plan that has an effect that unifies with the open condition.

In addition to satisfying open conditions, the planner resolves causal threats, which occur when the effect of some step s_t negates a causal link's condition e relating steps s_1 and s_2. Step s_1 establishes some condition e in the world that s_2 relies on for execution. But step s_t might occur after s_1 and before s_2, causing e to become false in the world and jeopardizing s_2's ability to succeed. We can repair causal threats by temporally ordering s_t before s_1 or after s_2.

Anticipating the player. Narrative plans lay out the entire action sequence to be performed during a storytelling session. Planning structures are advantageous for two reasons:

- Because planners lay out the entire expected sequences of events beforehand, they can omit planned actions that don't contribute to the story's coherence and outcome.
- A system can analyze the plan for points of possible failure due to unpredictable and interactive user behaviors.

Narrative plans lay out the entire action sequence to be performed during a storytelling session.

A narrative mediation system analyzes the story plan's causal structure to determine all possible exceptions that can occur during the narrative. For every possible exception, it generates an alternative story plan, beginning at the point of exception. This process results in a tree of story plans (a narrative mediation tree) in which each plan represents a complete story line—from the initial story world state or the exception causing the deviation from the parent to the conclusion. The system builds the narrative mediation tree before the interactive narrative session begins executing. This tree guides the storytelling system's interactive execution.

The system-controlled characters execute the generated script verbatim, while users can execute any legal action at any time regardless of the script. Indeed, the user is likely unaware of the story line or even how to remain consistent with the script. If the user performs an action that isn't part of the generated script, the system looks up the action to determine whether it's an exception. If it is, the system seamlessly switches to the appropriate alternative story plan and begins executing it immediately.

To prevent the narrative mediation tree from growing infinitely large, the system intervenes in some user actions. Intervention involves surreptitiously replacing a user action with a similar action, or failure mode, with different effects. The system maintains a list of failure modes for each possible user action. For example, the

Linear Narrative Generation

Narrative mediation generates interactive narrative experiences for the user by generating a linear narrative experience and then determining all the ways the user can deviate from that experience. A linear narrative generator—a narrative planner—is recursively called to generate alternative, contingency narratives. Several approaches to linear narrative generation exist.

Tale-Spin was one of the first linear narrative generators.[1] It used an inference rule engine to determine what the story-world characters would do in a noninteractive system. The inference rule approach has the same disadvantages as the autonomous agents approach to interactive storytelling in that it can't guarantee a coherent narrative.

The Universe storytelling system used hierarchical planning to decompose high-level plot fragments to generate more coherent narratives.[2]

More recently, Fabulist applies more formalized causal-dependency planning techniques to narrative generation.[3] The Fabulist narrative generator's use of causal dependencies is consistent with the use of causal links in narrative mediation. Fabulist generates narrative plans that aren't only coherent but also support character intentionality—the perception that story-world characters have goals and intentions that are distinct from the narrative plan's outcome (goal state).

Assuming that narrative mediation is at least as powerful as the conventional story graph representation of branching narratives and that a generative approach can partially mitigate the effort and complexity of authoring branching narratives by hand, we must look at the effort required to use a linear narrative generator such as those described earlier in this sidebar. A linear narrative generator can't operate in a vacuum; it requires knowledge about the story world in which the narrative is set. The more knowledge the linear narrative generator has, the greater the number of narrative content variations it can generate. (In other words, one can express a linear narrative generation system's creativity as the size of the set of potential narratives of value that the system can explore.) The more knowledge about a story-world domain that the generation algorithm can manipulate, the greater the search space will be.

Tale-Spin requires a large repository of common-sense knowledge in the form of inference rules. Universe and Fabulist require a library of plan operators and schemata (referred to as a domain theory), an initial world description, and a goal or outcome description. A domain theory describes in abstract terms all possible actions that story-world characters can perform. For all systems, the amount of knowledge required might be greater than the length of the linear narrative generated. However, the advantage of these knowledge bases is that they let a linear narrative generator generate multiple distinct narratives from the same knowledge base. Furthermore, for a technique such as narrative mediation, the system uses the same knowledge recursively over and over as it generates each branch in a branching narrative as a linear narrative. For branching narratives that are long or have a high branching factor, the amount of generated content can quickly exceed the size of the story-world domain authoring effort.

References

1. J. Meehan, "The Metanovel: Writing Stories by Computer," doctoral dissertation, Dept. of Computer Science, Yale Univ., 1976.
2. M. Lebowitz, "Story-Telling as Planning and Learning," *Poetics*, vol. 14, 1985, pp. 483-502.
3. M.O. Riedl, "Narrative Planning: Balancing Plot and Character," doctoral dissertation, Dept. of Computer Science, North Carolina State Univ., 2004.

user action *shoot* has failure modes *shoot-and-miss* and *gun-jam*. When the system determines that intervention is the best policy for handling an exception, it selects for execution a failure mode whose effects don't threaten the plan. An exception that is intervened with doesn't necessarily require an alternative story plan because the system preserves the original story plan's causal structure. A system also uses intervention when it can't replan for an exception. This often occurs when the user inadvertently destroys a resource (or a main character) that's necessary later in the story.

Example. A simple example interactive story about an inside-job bank robbery illustrates the narrative mediation technique. The narrative mediation process's end result will be a branching story covering all contingencies based on actions the user can perform. If the user performs no unanticipated actions, the story plays out as follows.

In the story, the user plays Sam, a night guard at a bank. The bank has a vault containing a large amount of gold. One night when Sam is on duty, the bank owner comes into the bank, retrieves his vault key from his office, unlocks and opens the vault, and begins removing the gold. Suspicious, Sam takes action to stop the bank owner from removing the gold.

The system automatically generates the story from a set of relatively simple input parameters: the initial story world state, a description of the outcome (the goal), and a domain theory that describes in general terms the actions that can be performed in the story world (such as open vault, pick up, and shoot). In this case, the initial story world state contains propositions describing the following situation:

- the bank owner is away from the bank;
- the bank vault is closed and locked;
- Sam has a key to the vault and a gun;
- a second vault key is in the office; and
- the gold is in the vault.

The outcome is that the bank owner has the gold, but he's also dead. The planner generates the following plan:

1. The bank owner enters the bank and goes into his office.

2. The bank owner picks up a key to the vault.
3. The bank owner goes to the vault.
4. The bank owner unlocks and opens the vault with the key.
5. The bank owner takes the gold out of the vault.
6. Sam, recognizing that a theft is in progress, shoots the bank owner before he can escape.

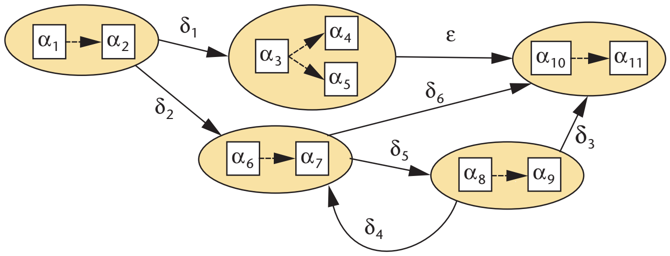
Of course, the player can perform many basic actions that threaten the story's coherence. For example, the player might remove the bank owner's key from his office before the bank owner arrives, shoot the bank owner before he commits the crime, or open the vault before the bank owner gets to it. (The result of this last action conflicts with step 4. Depending on the implementation details, this exception will either cause plan failure or execution will look strange because the bank owner will proceed to unlock the vault door when it's clearly already open.)

The narrative mediation system analyzes the story plan for any player action that can threaten the plan. That is, if the player were to perform action s_t during a particular interval—that is, insert the action into the plan in a particular interval—the plan would be flawed because of a causal threat, as we described earlier. For every possible causal threat, the system generates an alternative story plan that repairs the threat and thus achieves the outcome differently. If the player removes the bank owner's key, the bank owner will have to procure the key from the player. If the player opens the vault door prematurely, the bank owner will walk straight into the vault. If the player attempts to shoot the bank owner before he commits the crime, the system will intervene with one of the failure modes described earlier. The result of this recursive process is a narrative mediation tree, which is reported in greater detail elsewhere.[2]

Narrative mediation and story graphs

The narrative mediation technique demonstrates that any system that can generate a linear narrative plan (with causal annotation) and has a replanning capability can generate interactive narrative. However, many interactive narratives are expressed as story graphs; the use of branching story graphs is an intuitive way of expressing interactivity in storytelling. So, is narrative mediation's expressive power at least as powerful as the story graph representation? If it is, we can use narrative mediation to generate any branching narrative structure that can be represented as a story graph. Such an answer would conclusively demonstrate that we can apply linear narrative generation (see the related sidebar for a discussion of various approaches) to the branching narrative generation.

A branching story structure is a story graph—a directed graph of nodes connected by arcs representing user choices. Every possible path through the graph represents a story that can be told to the user. The number of arcs in a particular path in the branching story graph limits the user's sense of control over the story's development. Figure 1 shows an example story graph. The system starts out noninteractively with system-controlled characters performing actions α_1 and α_2. The user then chooses to perform action δ_1 or δ_2. If the user chooses δ_1, system-controlled characters perform actions α_3, α_4, and α_5. The system takes an ε-transition in the absence of any user action.



1 Story graph interleaving noninteractive story content (circular nodes) with decision point transitions.

The idea behind narrative mediation is to generate a linear story structure representing the best story that the system can tell. The linear story structure includes actions that the interactive user should perform interleaved with actions that system-controlled characters perform. If a user performs an exception, the system can either

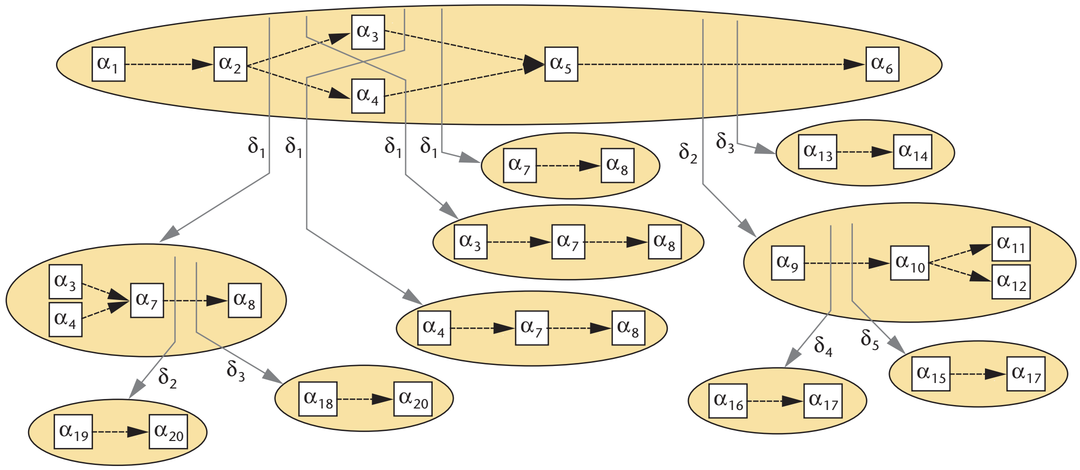
- intervene—that is, prevent the exceptional action from interfering with the story structure; or
- accommodate the action—that is, incorporate the exceptional action into the story and generate a new linear story structure that isn't threatened by the exception.

Figure 2 on the next page shows a narrative mediation tree with accommodated exceptions. The node at the top of the graph represents the best linear story that the system can tell. Both system-controlled characters and the user perform actions α_1 through α_6 (the system doesn't distinguish between them at generation time). Actions δ_1, δ_2, and δ_3 are exceptions. The figure shows that many arcs for the δ_1 exception exist because actions α_3 and α_4 in the node are unordered relative to each other. Thus, which arc the story follows and consequently which alternative story the system presents to the user depends on the actions executed prior to the exception. To gain additional opportunities to exert control, the user can perform exceptional actions at any time, regardless of whether the linear story includes a user action.

Narrative mediation generates multiple contingent story structures, making interactive narrative generation possible. However, we must prove that narrative mediation is at least as expressive as a system using the more conventional story graph. Thus, for any possible story graph, an equivalent narrative mediation tree must exist. If narrative mediation is at least as expressive as story graphs, one can use a linear narrative generation system to generate branching narratives through a process such as narrative mediation.

Proof definitions

Several definitions are necessary for the proofs. We assume that all graph structures used in the proof use a basic, partially ordered plan structure to represent temporally ordered actions.



2 Narrative mediation tree with accommodated exceptions.

Definition 1: Partially ordered plan. A partially ordered plan is a tuple $\prec A, O \succ$ such that A is a set of actions performed in the story world and O is a set of ordering constraints of the form $\alpha_i < \alpha_j$, where $\alpha_i, \alpha_j \in A$.

Figure 3 is an example of a partially ordered plan. The nodes are actions that story-world characters will perform, and the dashed arrows represent ordering constraints. Actions that are unconstrained with relationship to each other can execute in parallel or in an arbitrary sequential order.

A story graph G is a set of partially ordered plans $P \in G$ representing the noninteractive sequences of events. Story branches, representing decision points, connect the nodes. After every partially ordered plan executes, a finite number of actions $D \in G$ exists, some of which are possible user choices.

Definition 2: Story branch. A story branch in story graph G is a tuple $\prec p_1, C, \delta, p_2 \succ$ such that $p_1, p_2 \in P$, $\delta \in D$, and C is an implementation-specific set of applicability criteria. The story branch indicates that p_1 is temporally ordered before p_2 and that δ is an action executed by the user to cause transition from p_1 to p_2.

Definition 3: Story graph. A story graph is a tuple $\prec P, D, \Lambda, B \succ$ such that P is a set of partially ordered plans, D is a set of user actions, Λ is an action library such that for all $p_i \in P, A(p_i) \subseteq \Lambda$, and B is a set of story branches such that $C(b) = \varnothing$ for all $b \in B$.

The story-graph representation doesn't disallow cycles. A cycle in a story graph means there is a point in the story where the user can make a decision that causes a previous portion of the story to repeat. An acyclic story graph, however, explicitly prohibits cycles, implying that a story has a finite duration. A story tree is a special type of acyclic story graph that doesn't reuse any node; each story branch terminates in a unique node.

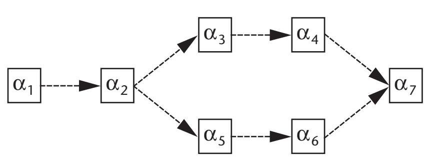
The narrative mediation tree is the primary structure used by narrative mediation. Although the data structure is superficially similar to a story graph, narrative mediation uses the story nodes in different ways. Instead of being a short sequence of actions between decision points, a story node in narrative mediation is meant to represent the entire story if the user doesn't interfere.

Definition 4: Narrative mediation tree. A narrative mediation tree is a tuple $\prec P, D, \Lambda, B \succ$ such that P is a set of partially ordered plans, D is a set of user operations, Λ is an action library such that $D \subseteq \Lambda$ and for all $p_i \in P, A(p_i) \subseteq \Lambda$, and B is a set of story branches without cycles such that for all $b \in B$, $C(b) \subseteq A(p_1(b))$, and $C(b)$ is a prefix of $p_1(b)$.

The applicability criteria of b is the sequence of actions forming a prefix of b's originating node. $C(b)$ is the history of the actions that have been executed in this node. The system uses the history to uniquely identify an arc if several story branches with the same exceptional user action originate from the same node.

Proofs

If narrative mediation trees' expressive power is at least as powerful as that of acyclic story graphs, narrative mediation can generate any interactive branching story line that systems that implement acyclic story graphs use. To prove this, we show that the set of all acyclic story graphs is a subset of the set of all narrative mediation trees. The subset relationship is true if an algorithm exists that transforms any arbitrary acyclic story graph into an equivalent narrative mediation tree representation. For any graph G_1 of one form of representation to be equivalent to another graph G_2 of another form of representation (for example, for an acyclic story graph to be equivalent to a narrative mediation tree), the set of all paths through G_1 (the set of all stories that can be told) must be a subset of the set of all paths through G_2. A path through either structure includes the system-controlled character actions interleaved with user actions. For example $\{\alpha_1, \alpha_2, \delta_2, \alpha_6, \alpha_7, \delta_5, \alpha_8, \alpha_9, \delta_3, \alpha_{10}, \alpha_{11}\}$ is one path through the story graph in Figure 1.



3 A partially ordered plan (definition 1).

Simple proofs can establish the following facts. (Length constraints prohibit the inclusion of all the proofs here.)

The set of all acyclic story graphs is a subset of all story graphs. By definition, acyclic story graphs are story graphs.

The set of all acyclic story graphs is equal to the set of all story trees. We can transform an acyclic graph into a tree by duplicating nodes with incoming arcs. Starting with leaf nodes and working back to the root, for every node with more than one parent, duplicate the subtree with the child node as root $n - 1$ times where n is the number of parents, and make each parent point to a different subtree.

The set of all narrative mediation trees isn't a subset of the set of all story graphs. Narrative mediation lets user actions execute concurrently with system-controlled character actions whereas story graphs require interleaving user and system actions.

We prove this by contradiction. Suppose the set of all narrative mediation trees is a subset of the set of all story graphs. Then, we can represent any specific instance of narrative mediation trees as equivalent story graphs. Let G_m be a narrative mediation tree such that the ordering of two actions, s_u and s_a, is underconstrained and that s_u and s_a could occur concurrently. Furthermore, suppose that s_u is an action for the user to execute and s_a is an action for a system-controlled character to execute. Definitions 1 through 3 imply that any path through a story graph must consist of sequences of system-controlled character actions interspersed with user actions corresponding to decision points. Therefore, we can't represent G_m as a story graph. Therefore, the set of all narrative mediation trees isn't a subset of the set of all story graphs.

To be thorough, we could add ordering constraints between s_u and s_a to create a valid story graph representation of G_m (for example, we could constrain s_u to strictly occur before s_a). In this case, however, the set of all paths through G_m and the resultant story graph aren't guaranteed to be the same, especially considering that narrative mediation trees with unordered actions, such as that shown in Figure 2, can have different branches depending on whether the user action was performed before or after concurrent system-controlled character actions are completed.

The set of all narrative mediation trees isn't a subset of the set of all acyclic story graphs. The proof of this is an argument identical to the proof by contradiction in the previous case because acyclic story graphs also disallow concurrent user actions and system-controlled character actions.

Because story graphs can have loops, they can have infinite-length paths; narrative mediation doesn't allow infinite stories.

The set of all story graphs isn't a subset of the set of all narrative mediation trees. Because story graphs can have loops, they can have infinite-length paths; narrative mediation, on the other hand, doesn't allow infinite stories. (Cyclic story paths result in infinite-depth branches in a narrative mediation tree. In practice, we don't anticipate that users will attempt to loop forever, justifying our limitation of tree depth. Interventions can help enforce depth limits.)

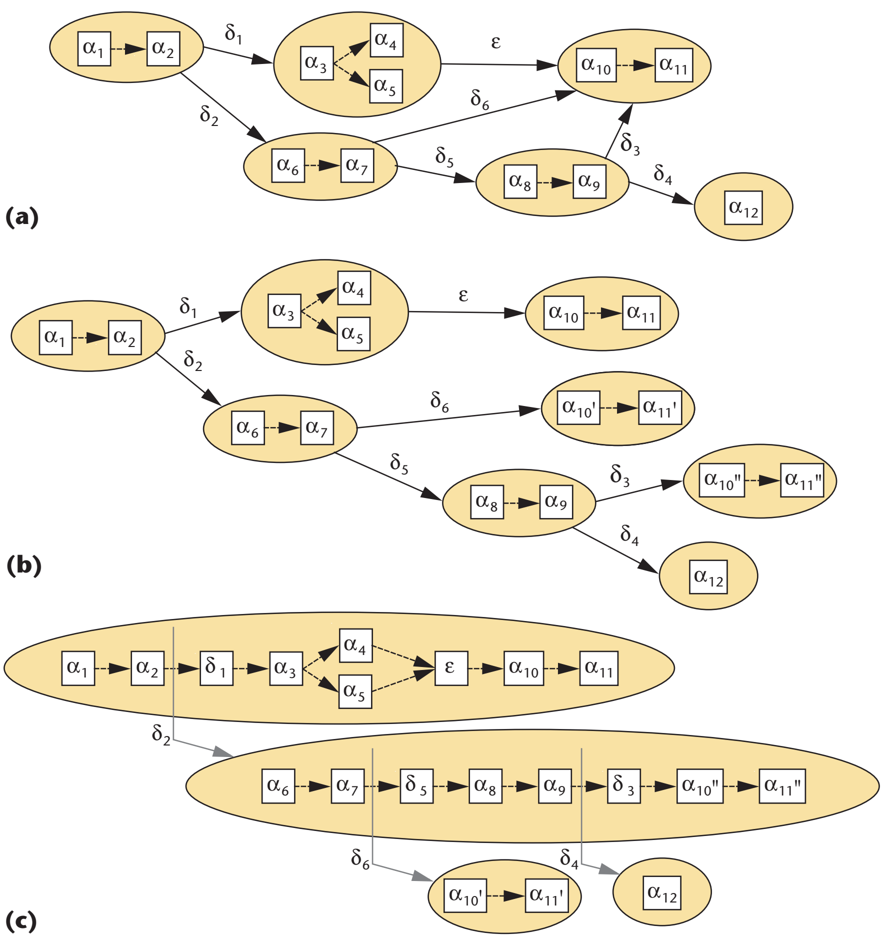
We prove this by contradiction also. Suppose the set of all story graphs is a subset of the set of all narrative mediation trees. This means that we can represent any specific story graph instance as an equivalent narrative mediation tree. Suppose G_s is a story graph such that $B(G_s)$ has a cycle. Definition 4 states that narrative mediation trees can't have cycles. Let p be the longest path through a narrative mediation tree equivalent to G_s. According to the pumping lemma, any path through G_s that includes the cycle can be pumped an arbitrary number of times until it's longer than p. Therefore, no narrative mediation tree exists that's equivalent to G_s. Therefore, the set of all story graphs isn't a subset of the set of all narrative mediation trees.

The set of all acyclic story graphs is a subset of the set of all narrative mediation trees. The key to proving the hypothesis, however, is proving that the set of all acyclic story graphs is a subset of the set of all narrative mediation trees. To prove this, we provide an algorithm that can transform any arbitrary instance of an acyclic story graph into an equivalent narrative mediation tree. Let G_a be an acyclic story graph. Figure 4 (on the next page) is the algorithm that transforms an arbitrary instance of an acyclic story graph into a narrative mediation tree, G_m. The proof that the set of all acyclic story graphs is a subset of the set of all narrative mediation trees depends on the proof of the transformation algorithm's correctness. That is, given an acyclic story graph, the transformation algorithm always produces a narrative mediation tree that's equivalent to the original acyclic story graph.

The proof of correctness of the algorithm in Figure 4 relies on the fact that a node in G_m contains both system-controlled character actions and user actions, representing one path through G_a. Line 5 creates a new node that's a concatenation of two nodes in G_a plus the user action that transitions between the two nodes. Line 11 removes the old nodes from G_m and adds the new amalgamated node. As a result, G_m has a minimal number of nodes that represent the maximum-length subpaths in G_a. Whether or not G_m is equivalent to G_a relies on the positioning of the arcs in G_m. If two arcs originate in a single node in G_a, then in G_m, the user action of one of the arcs must be part of a partially ordered plan in a story node in G_m while arcs originating in that node in G_m represent the other user actions. Line 5 incorporates a user action into a new node. Line 9 identifies the alternative arcs in G_a (because the arc originates from the

Let G_a be an acrylic story graph.	
1. Let $G_t = \prec P, D, \Lambda, B \succ$ be a story tree that is equivalent to G_a.	[Make a story tree out of the acyclic graph.]
2. Let $D' = B' = \emptyset$. Let $B'' = B$. Let $P' = P$. Let $\Lambda' = \Lambda$.	
3. If $B'' = \emptyset$, then halt and return $G_m = \prec P', D', \Lambda', B' \succ$. Otherwise, Let $b'' = \prec p_1, \emptyset, \delta, p_2 \succ$ be an arc in B''.	[Loop until B'' is empty. Otherwise, pick any story branch.]
4. Let $L \subseteq A(p_1)$ such that $\alpha_i \in L \rightarrow \prec \alpha_i, \alpha_j \succ \notin O(p_1)$ for all $\alpha_i, \alpha_j \in A(p_1)$. Let $F \subseteq A(p_2)$ such that $\alpha_l \in F \rightarrow \prec \alpha_j, \alpha_l \succ \notin O(p_2)$ for all $\alpha_l, \alpha_j \in A(p_2)$.	[Collect all last steps from the 1st node and all first steps from the 2nd node in branch.]
5. Let $p' = \prec A(p_1) \cup A(p_2) \cup \{\delta\}, O(p_1) \cup O(p_2) \cup \{\prec l, \delta \succ$ for all $l \in L\} \cup \{\prec \delta, f \succ$ for all $f \in F\} \succ$. Let $\Lambda' = \Lambda' \cup \{\delta\}$.	[Create a new plan node that merges the two nodes.]
6. For all $b = \prec p_i, H, \delta_k, p_j \succ \in B'$ such that $p_i = p_1$, $B' = B' - \{b\} \cup \{\prec p', H, \delta_k, p_j \succ\}$.	[Update any story branches in mediation tree that originate from p_1.]
7. For all $b = \prec p_i, H, \delta_k, p_j \succ \in B'$ such that $p_j = p_1$ or $p_j = p_2$, $B' = B' - \{b\} \cup \{\prec p_i, H, \delta_k, p' \succ\}$.	[Update any story branches in mediation tree that terminate in p_1 or p_2.]
8. $B'' = B'' - \{b''\}$.	
9. For each $b = \prec p_i, \emptyset, \delta_k, p_j \succ \in B''$ such that $p_i = p_1$, $B'' = B'' - \{b\}$, $B' = B' \cup \{\prec p', A(p_1), \delta_k, p_j \succ\}$, and $D' = D' \cup \{\delta_k\}$.	[Other branches coming out of p_1 become exceptions.]
10. For each $b = \prec p_i, \emptyset, \delta_k, p_j \succ \in B''$ such that $p_i = p_2$, $B'' = B'' - \{b\} \cup \{\prec p', \emptyset, \delta_k, p_j \succ\}$.	[Update any story branches in story tree that originate from p_2.]
11. $P' = P' - \{p_1, p_2\} \cup \{p'\}$.	[Add new node to mediation tree.]
12. Go to step 3.	

4 Algorithm for transforming an arbitrary instance of an acyclic story graph into a narrative meditation tree.



5 Identical representations of a branching story. (a) The initial acyclic story graph representation for the branching story. (b) The story tree representation for the branching story. (c) The narrative mediation tree representation for the branching story.

same node in G_a) and creates a new arc in G_m originating from the new node.

The proof of correctness also relies on the correct determination of the applicability constraints (line 9) for all story branches in the resulting narrative mediation tree. We can prove the correctness directly, but this is beyond this article's scope. Consequently, G_a and G_m are equivalent. Therefore, we can transform any arbitrary acyclic story graph into an equivalent narrative mediation tree. Therefore, the set of all acyclic story graphs is a subset of all narrative mediation trees.

Figure 5 shows three identical representations of the same branching story:

- an acyclic story graph representation of a branching story;
- a story tree representation where the story tree is derived by duplicating nodes in the acyclic graph that have more than one parent so that each parent in the tree has its own unique clone of the original child node (because of how we duplicate certain nodes, actions α_{10}, α_{10}', and α_{10}'' in the story tree representation are all identical, as are α_{11}, α_{11}', and α_{11}''; and
- a narrative mediation tree derived from the story tree using the algorithm in Figure 4.

Conclusion

As the proofs demonstrate, narrative mediation is at least as powerful as interactive narrative systems with acyclic branching stories. Narrative mediation trees can represent interactive stories that story graphs can't because they let users perform actions concurrently with system-controlled character actions. However, because narrative mediation trees can't have cycles, they can't represent cyclic story graphs. Figure 6 shows the relationship between story graphs and narrative mediation trees.

The driving force behind our research is the idea that interactive storytelling systems create the possibility of new computer game genres that are cognitive in nature. Games based on interactive storytelling techniques let players exert a degree of perceived control over their characters' fate by letting them make decisions and perform actions that directly impact the story plot and possibly the story outcome.

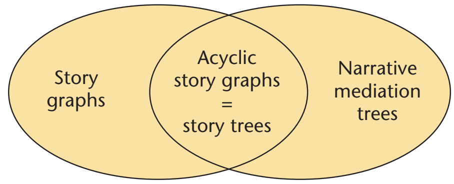
Many possible benefits exist to considering linear narrative generation as a part of the process of generating branching narratives.

First, regardless of the complexity of a branching narrative structure, in a single interactive storytelling system run, the user's experience from his or her perspective is one of a linear narrative. To determine whether the user's experience is compelling or engaging, one must analyze individual paths through the branching story structure. If the system generates each desired path as a separate linear narrative and aggregates it into a narrative mediation tree, we can incorporate any given path's "goodness" into the constraints of the linear narrative generation process itself. Fabulist (see the sidebar) does this for coherence and character intentionality.

Another benefit is the long tradition in human history of authoring linear narrative. It might be possible to directly leverage human story authors' knowledge and expertise to develop better models of good story sequences as well as better computational models of story generation without worrying about branching and interactivity. Gordon et al., for example, describe a technique for authoring outcome-driven branching stories that starts with a library of complete, linear narrative vignettes that are manually assembled into a branching structure.[1] ■

Acknowledgments

The US National Science Foundation Career Award no. 0092586 supported the work presented here. Statements and opinions expressed don't necessarily reflect the US government's position or policy, and no official endorsements should be inferred.



6 **Relationship between story graphs and meditation trees.**

References

1. A. Gordon et al., "Branching Story Lines in Virtual Reality Environments for Leadership Development," *Proc. 16th Innovative Applications of Artificial Intelligence Conf.*, AAAI Press, 2004, pp. 844-851.
2. M.O. Riedl, C.J. Saretto, and R.M. Young, "Managing Interaction between Users and Agents in a Multiagent Storytelling Environment," *Proc. 2nd Int'l Joint Conf. Autonomous Agents and Multiagent Systems*, ACM Press, 2003, pp. 741-748.
3. R.M. Young, "An Architecture for Integrating Plan-Based Behavior Generation with Interactive Game Environments," *J. Game Development*, vol. 1, no. 1, 2004.
4. R. Aylett, "Emergent Narrative, Social Immersion, and 'Storification,'" *Proc. 1st Int'l Workshop Narrative and Interactive Learning Environments*, 2000.
5. M. Cavazza, F. Charles, and S.J. Mead, "Planning Characters' Behavior in Interactive Storytelling," *J. Visualization and Computer Animation*, vol. 13, 2002, pp. 121-131.
6. J. Bates, "Virtual Reality, Art, and Entertainment," *Presence: J. Teleoperators and Virtual Environments*, vol. 1, no. 1, 1992, pp. 133-138.
7. P. Weyhrauch, "Guiding Interactive Fiction," doctoral dissertation, School of Computer Science, Carnegie Mellon Univ., 1997.
8. B. Magerko et al., "AI Characters and Directors for Interactive Computer Games," *Proc. 16th Innovative Applications of Artificial Intelligence Conf.*, AAAI Press, 2004, pp. 877-883.
9. A. Lamstein and M. Mateas, "Search-Based Drama Management," *Proc. 2004 AAAI Workshop on Challenges in Game Artificial Intelligence*, AAAI Press, 2004, pp. 103-107.
10. M. Mateas and A. Stern, "Structuring Content in the Façade Interactive Drama Architecture," *Proc. 1st Ann. Conf. Artificial Intelligence and Interactive Digital Entertainment*, AAAI Press, 2003, pp. 93-98.
11. J.S. Penberthy and D. Weld, "UCPOP: A Sound, Complete, Partial-Order Planner for ADL," *Proc. 3rd Int'l Conf. Knowledge Representation and Reasoning*, Morgan Kaufmann, 1992, pp. 103-114.

Mark Riedl *is a research scientist at the University of Southern California's Institute for Creative Technologies. His research interests include AI for entertainment, education, and training, particularly involving automated narrative generation, interactive narrative, and intelligent virtual agents. Riedl has a PhD in computer science from North Carolina State University. Contact him at riedl@ict.usc.edu.*

R. Michael Young *is an assistant professor of computer science at North Carolina State University. His research interests are in AI, planning algorithms, discourse generation, and models of interactive narrative. Young has a PhD in intelligent systems from the University of Pittsburgh. He was awarded an NSF Career Award in 2000. Contact him at young@csc.ncsu.edu.*