

A Platform for Symbolically Encoding Human Narratives

David K. Elson and Kathleen R. McKeown
Columbia University Department of Computer Science
New York City

Abstract

We present SCHEHERAZADE, a foundational platform for narrative intelligence that formally represents stories. The system draws upon prior theoretical work on the morphology of narrative without imposing a particular narrative domain or task *a priori*. Instead, while keeping *narrative semantics* separate and immutable, it provides a framework for the tools that build upon it to define world knowledge on a per-task basis. The system models semantics such as timelines, states, events, characters and goals, and can detect thematic patterns in both the deep content of the story (i.e., the *fabula*) and the manner of the story's telling (the *sjuzhet*). We also present a sample tool that utilizes the platform to assist users who are unfamiliar with narratology to symbolically encode stories into this representation. A formative evaluation of this tool shows that technical and non-technical users can successfully encode a traditional fable using its graphical interface.

Introduction

As a structural technique for communicating information to others, and deriving meaning for ourselves, humans invoke narrative form in domains reaching well beyond fictional storytelling. The prevalence of narrative has placed it at the intersection of many fields. While cognitive scientists seek the “mental operations that enable readers to make the leap from symbols on a page to elaborate models of narrative worlds” (Gerrig & Egidi 2003; Bruner 1991), AI researchers explore the algorithmic nature of storytelling and build systems that interact with humans on narrative terms (Mateas & Sengers 1999).

Narratologists distinguish between content and presentation layers of narrative. The content layer, sometimes called the *fabula*, describes the world of the story and all the causally related actions that occur. As (Ryan 1991) puts it, a narrative consists of a chronologically ordered sequence of *states*, which include propositions about the world at some point in time, and *events*, which serve as transitions between states. For example, a character may be at one *location* in State 0, and a different location in State 1, if it moves during the transition. The presentation layer, or *sjuzhet*, covers the

selection, ordering and description of *fabula* events as they are told in the linear progression of the story. (Bal 1997) calls this *focalization*, and also distinguishes a third layer, *text*, in which a separate *narrator* brings the story into an observable medium such as prose or film.

In this paper, we present SCHEHERAZADE, a computational platform for representing narratives. We have designed it to implement this conventional narrative theory, independent of a particular narrative domain or task in narrative intelligence. Specifically, unlike other modeling approaches, SCHEHERAZADE reasons over symbols related to narrative structure rather than the mechanics of the story-world itself (that is, inference over world knowledge). Freed from the constraints of full understanding, the system models narrative devices which are often overlooked, such as the interplay between the *fabula* and *sjuzhet* of a story, and *diegetic timelines*, which extend character goals and beliefs across the time dimension.

SCHEHERAZADE works in conjunction with *higher-level tools* that leverage the representation to address problems in narrative intelligence. We will use this term to refer to future systems, built upon this implemented foundation, that apply the narrative framework toward some practical end. We envision SCHEHERAZADE empowering a range of tools for tasks such as co-construction, understanding, and in particular, corpus analysis, by extracting complex symbolic features for automatic learning. (Thus, some higher-level tools may interface directly with users, where others focus on back-end analysis.) In contrast to standard computational linguistics, such an approach enables experiments that bridge users to the semantics of story understanding, and allows machine learning tools to find thematic patterns which are not evident in surface text. To date, we have implemented one such tool, which defines a story-world and provides a GUI so that users can easily encode stories.

Related Work in Story Analysis

Prior work in linguistic story analysis descending from (Labov & Waletzky 1967) has worked down from the surface level, annotating words and clauses with their symbolic roles and reconstructing the original temporal sequence of the *fabula*. Other researchers have distanced themselves further from the surface medium in order to reach broader symbolic patterns. Propp's groundbreaking study (Propp 1969)

reduces a corpus of Russian folktales to 31 common “functions,” each defined in the least abstract terms that satisfy the covered instances. The function *The hero is pursued*, for example, is drawn from scenes that span a range of scenarios from flying after the hero to gnawing at a tree in which the hero takes refuge. It is no accident that Propp and the structuralists who followed him drew from folktales and myths. In these oral traditions, the story’s telling changes from generation to generation, but the essential and timeless elements of the *fabula* remain intact.

On the computational side, most work in natural language processing aims to extract meaning from text, and stories have long been a domain of interest for total or partial understanding using models such as scripts and plans (Schank & Riesbeck 1981), conceptual graphs (Graesser, Lang, & Roberts 1991), and first-order predicate calculus (Mueller 2003). A central challenge of this approach is to model not only what is said, but what is implied, or deliberately left unspecified. A storyteller rarely reports *every* assertion about *every* state of the story world, and instead assumes that the listener shares a similar model of world knowledge in which some assertions are inferred from others (McKoon & Ratcliff 1992). This world knowledge may differ by domain. When reading a fairy tale, for example, we might assume that certain animals have the ability to talk (Ryan 1991, 48-60).

Rather than aim for complete automatic interpretation of a *fabula* from a text, recent work has focused on encoding enough narrative insight to more formally investigate the human narrative instinct. One contemporary annotation scheme for analyzing sets of narratives allows semantic units to “emerge” from clusters of text which convey the same basic content, regardless of the degree of lexical overlap (Pasonneau, Goodkind, & Levy 2007). Others have used this partial-modeling approach for building systems that classify the completeness of children’s retellings (Halpin, Moore, & Robertson 2004), interact with children to encourage narrative proficiency (Cassell 2004) or categorize certain aspects of narrative such as a listener’s likely affectual response (Alm & Sproat 2005).

Design and Representation

SCHEHERAZADE serves as a platform for experiments and practical applications that build upon symbolic representations of narratives. It can encode, store, and analyze a single story or a corpus of stories for whatever purpose is dictated by a higher-level tool. To ensure this versatility, SCHEHERAZADE is robust enough to handle narratives in diverse genres and styles, while still formalizing a common structural framework. To this end, the system is not tied to a particular domain, but rather is modular with respect to knowledge management. Specifically, it separates out *narrative semantics* from both *world knowledge* and the particular content of individual stories. In this section, we describe the relationship between these three types of objects, which are illustrated in Figure 1 and can be summarized as follows:

1. **Narrative semantics.** At its core SCHEHERAZADE represents stories with symbols representing a closed set of

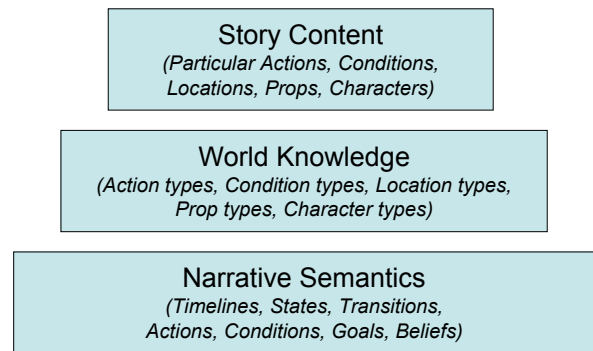


Figure 1: Three classes of knowledge are distinguished by SCHEHERAZADE, each of which applies the one that appears beneath.

abstract structural elements, and handles the relationships between them.

2. **World knowledge.** Symbols in this class represent the domain in which the story takes place, e.g., the types of objects which can exist, and the types of actions that characters can take.
3. **Story content.** To represent actual stories, the system stores instances of the world-knowledge frames (such as particular characters and actions that they take).

By storing all the symbols in a structural framework, SCHEHERAZADE constructs the *fabula* of the story; simultaneously, by preserving the order in which the symbols arrive (which is the telling of the story), it constructs a record of the *sjuzhet*. Both of these are available to higher-level tools. For example, one tool could extract features from the *fabula* and train a learning algorithm to detect “happy endings” vis-a-vis some character’s goals, or model the affectual state of a reader at each point in the *sjuzhet*.

Narrative Semantics

The concepts that SCHEHERAZADE formalizes in this class are immutable, so they provide a consistent structural framework across individual stories and domains. This uniformity provides the basis for its analysis and comparison tools. These “narrative primitives” include:

- **State.** A state is a single moment in the world of the story. It is assigned a time index. Depending on the situation, this time index may correspond closely to absolute time values (e.g., characters only have until sundown to accomplish some task), or only ordered relative to one another. States may also be declared as representing spans of time, either bounded (between two time indexes) or unbounded (extending from the indefinite past or into the future).
- **Physical object.** Three classes of physical objects are distinguished:
 - *characters*, which can serve as agents for effecting change;
 - *props*, which have no autonomy as characters do; and

- *locations*, which characters can inhabit.
 - **Condition.** A condition is a property of a state that describes the physical or psychological status of an object or character during the point in time associated with the state.
 - **Transition.** A transition joins two states. The states need to be in the same timeline, but they do not need to be adjacent. Transitions serve as hosts to actions which take place during the time between the two states.
 - **Action.** An action is a process that takes a definite span of time. This span may be effectively instantaneous (such as snapping one’s fingers) or lasting over a longer period (such as traversing from one location to another). Because transitions can link nonadjacent states, actions might be “in progress” during other states. There are several types of actions:
 - *agent actions*, which are caused by characters (intentionally or not), and
 - *happenings*, which might happen *to* characters, but are not directly caused by them.
 - **Timeline.** A timeline is a set of states and connecting transitions, which in turn include actions, conditions, and references to physical objects. There are several types of timelines:
 - *Reality timeline.* There is one reality timeline for each story-world, and it depicts the chronological ordering of the actual *fabula*. (This assumes a reliable narrator; the assertions of unreliable narrators would populate a diegetic timeline, described next.)
 - *Diegetic timeline.* A diegetic timeline exists relative to a character within the story-world, by one of the following associations:
 - * *Goal.* A goal timeline includes the state (or action) that a character desires to occur.
 - * *Plan.* A plan timeline indicates what states and actions the character is intending to transpire, typically through its own actions.
 - * *Belief.* A belief timeline indicates a character’s internal conception about the nature of the story-world, including (but not limited to) the state in which the character holds the belief. In other words, a belief timeline may include beliefs about the past, present and future.
- Each of these timelines align to a *referent timeline*, which can be either the reality timeline or another diegetic timeline (to arbitrary levels of nesting). For example, a character may believe that another character believes something about the first character, or a character may have a plan that branches into a primary goal and a backup goal, depending on whether some intermediate action occurs.
- The diegetic timeline aligns with the referent timeline in one of two ways:
- * *Absolute* indicates that the time indexes in the diegetic timeline refer directly to those in the referent timeline,

regardless of *when* in the referent timeline the character has the goal, plan or belief, e.g., *The dog believed that when the owner came in, he had a treat.*

- * *Relative* indicates that the diegetic timeline measures time as a distance from the particular state in the referent timeline when the diegetic timeline is invoked, e.g., *The dog believed that his owner had a treat, and the owner would give him the treat in a few seconds.*
- *Repeated timeline.* This timeline represents sequences of states and transitions that, as a unit, occur at least once over some span of time in a referent timeline. These can model concepts such as, *For years the goose laid golden eggs, and the man sold them.*
- **Setting.** Each timeline includes a data structure which stores the declared frames of world knowledge that are valid for that timeline and all timelines that refer to it.

While this is a high degree of structural complexity, higher-level tools can choose to expose or hide whichever subtleties are appropriate on a per-task basis. For example, the graphical interface we discuss below, which allows users to encode original or *a priori* stories, currently exposes to them a simpler, less expressive formalism for entering goals and beliefs. Similarly, were SCHEHERAZADE to serve as a representational tool for an automatic story understanding system, the accuracy of such machine readings would closely depend on which elements of the model were used (as it would be easier to detect a list of characters than to parse diegetic timelines from surface text).

World Knowledge

Each higher-level tool first populates SCHEHERAZADE with world knowledge pursuant to the needs of its particular domain. The underlying system is agnostic to whether the tool hard-codes this knowledge, or gathers it from users or another external source. The facets of knowledge, which correspond to the narrative primitives described above, include hierarchies of prop types (e.g., *weapon*), action types (*firing a weapon*), condition types (*hungry*), location types (*building*), and character taxonomies (*human* versus *animal* and *mule*).

In each case, a facet is permitted to take parameters, which are validated against the other facets as well as the overall story each time the frame is instantiated. For example, a location type `foyer (<house>)` indicates that all foyers exist in the context of a house (though not all houses necessarily have foyers), and that users cannot indicate that a character is in a foyer without also indicating the house in which the foyer lies. (When the particular house is unknown or unimportant, one can define an “anonymous” instance of a frame that is only valid for the scope of the assertion, i.e., `foyer (new House ())` in Java-like syntax.) Thus, the world knowledge parameters instantiate the narrative semantics, while at the same time serving as frames to guide the content of particular stories.

This architecture for encoding world knowledge features *optional* complexity; that is, higher-level tools may pick and choose which areas of the world to encode, and the granularity of detail at which to encode them. A comprehensive

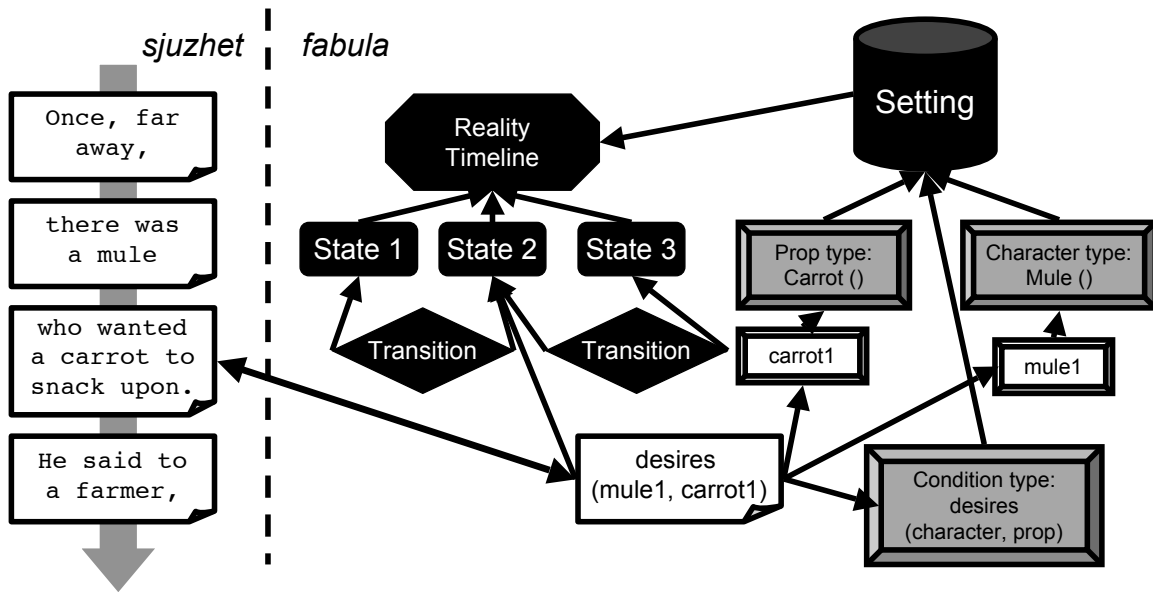


Figure 2: An example of the type of conceptual graph which SCHEHERAZADE uses to represent narratives. Narrative semantics, world knowledge and story content are in black, grey, and white shapes, respectively.

model of world knowledge is not necessarily required. For example, if multiple users are annotating the same *a priori* story, one might need to maximize inter-annotator agreement. In this case, a smaller set of available symbols ensures some degree of annotation overlap, so that the models can be compared. In the case of narrative authoring tools, on the other hand, users often desire a generous creative range, and comprehensive domain knowledge is appropriate. In short, SCHEHERAZADE is able to gracefully handle stories told with varying levels of abstraction. We are exploring the possibility of applying existing knowledge bases such as CYC (Matuszek *et al.* 2006) and WordNet (Fellbaum 1998).

SCHEHERAZADE also provides a framework for higher-level tools to attach certain types of semantic interpretation rules to the world-knowledge frames. These include hierarchical arrangement (such that `walk()` descends from `move()` and implies its supertype) and satisfaction rules (so that an instantiation of `die(<character>)` forbids the instantiation of actions in future states that have the character as an agent). However, these rules are optional, and the system only checks that a story makes sense (on a content level) where they are provided. This enables it to encode narrative devices, such as unreliable narrators and flashbacks, that are too complex to be expressed in representations that are based on total understanding of the story-world. The tradeoff is that, unlike approaches such as full- and partial-order planning (Riedl & Young 2005), SCHEHERAZADE is not sufficiently aware of the story’s semantics to compute a new state from a given state and an action; it cannot “solve” for a character’s plan to reach some goal. Rather, SCHEHERAZADE is better suited to address problems in narrative intelligence which reason over the thematic dimension of human narratives.

Story Content

Strongly-typed programming languages allow coders to first declare variables, and then define them according to the rules of their types. Similarly, SCHEHERAZADE provides a framework for higher-level tools to declare the frames of knowledge in their narrative domains, and then describe actual stories by instantiating them. The system populates the representation as it interprets a sequence of “story assertions” that it validates against both the hard-coded rules of narrative semantics and the customized rules of world knowledge. Symbolically encoding a story, then, is an iterative process of asserting a point and receiving acknowledgment that it can be assimilated on both the structural and content levels (where applicable).

SCHEHERAZADE assembles the symbols of all three facets of its knowledge – narrative semantics, world knowledge and story content – in a single conceptual graph. A simplified diagram of one such graph is shown in Figure 2. The black boxes are nodes representing narrative semantics, including states, transitions, and a timeline. The grey boxes are nodes representing world knowledge, including declarations of character types, prop types, and condition types. Finally, the white boxes are nodes representing story content, instantiating the world knowledge. This *story graph* represents the system’s understanding of the complete story at a certain point in the *sjuzhet*, which is itself modeled as a vector of assertions at the left of the figure.

Implementation

We have implemented SCHEHERAZADE in Java, so that it is modular and easily accessible to higher-level tools. The current state of our implementation features a three-layer architecture, which, from the bottom up, is as follows:

- **Conceptual graph kernel.** At the core is a tool for defining, building and searching over arbitrary conceptual graphs. The kernel accepts parameterized rules for the types of nodes, types of links, and types of inference which may figure in the graph. It also supports node attributes, frames nodes, and instance nodes, as well as atomicity and undo/redo when building graphs. A GUI allows for graph inspection and manipulation, and a query language allows higher-level classes to search for particular patterns of nodes or determine satisfaction of a pattern by a given set of nodes.
- **SCHEHERAZADE.** This layer implements the design parameters outlined in the previous sections, interpreting *sjuzhet* assertions into well-formed conceptual graphs similar to the one in Figure 2. Higher-level tools (and, in turn, users) do not have direct access to the graph; rather, this layer provides an API for both declaring facets of world knowledge and asserting the details of a particular story.
- **Sample tool with encoding interface and text generator.** Finally, we have implemented one such higher-level tool that, while separate from SCHEHERAZADE, applies it to a particular task. Specifically, this layer provides a graphical interface that allows users to symbolically encode narratives as story graphs. An animated timeline displays boxes and diamonds to represent states and transitions, respectively. Users can define characters, props, locations, actions and conditions, and assign them to the story-world. Rather than expose first-order predicates to the user, the interface includes a template-based natural language generator that creates a prose equivalent for any part of the story graph. The generator varies the tense, aspect and style of its output depending on the circumstances (for example, present tense when viewing a state, but past tense when summarizing the entire story). In this manner, the symbolic underpinnings of the representation, as well as the story graph itself, are hidden from the user.

Figure 3 shows the encoding tool as customized for the domain of Aesop’s Fables (relevant for the formative evaluation below). Along the bottom left is the graphical timeline, with an arrow indicating the transition currently being viewed. The banner at the top of the left panel summarizes, using the prose generator, the actions that take place during this transition. The middle portion of the left panel describes the current affairs of each character, including actions that take place during the present transition as well as actions that began in a previous transition and continue to occur. The right-hand panel compares the input and output of the tool: a natural-language story from which the user draws, and a “reconstructed story” that is procedurally generated from the entire story graph. The user continues to instantiate new symbols (using buttons such as “New Instant Action”) until the reconstructed story is sufficiently similar to the original story.

As we have described, the SCHEHERAZADE API allows higher-level tools to set and retrieve specific story elements. In addition, it offers a search module which finds occur-

rences of certain symbolic patterns in the story graph, and determines whether a given symbol satisfies some pattern. These queries, built from first-order predicates, can represent thematically significant content such as goal attainment: *there are two states, one occurring after the other; in the first state, a character has a goal state to achieve some condition X, and in the second state, the character has a positive association with X.* Using this tool, SCHEHERAZADE can identify the thematic features of a story graph, which can be compared to other stories or passed to machine learning systems for further analysis (see Future Work).

Formative Evaluation of Sample Tool

There are several measures of the quality of a narrative representation: its expressiveness (the range of stories it can encode), robustness (how well it handles partial or abstractly told stories), formality (how well its symbols lend themselves to formal analysis or automatic processing) and usability (how intuitive the representation is for collecting encodings from those not versed in narrative theory). As our near-term goal is to collect a corpus of story graphs, encoded by human subjects with our system, the last measure has been a particular focus. In this section, we will describe a small formative evaluation designed to determine whether SCHEHERAZADE as a platform, and the tool described above as an interface, are usable.

We designed our evaluation to be *task-oriented*: Rather than give users an open-ended opportunity to tell original stories, we asked them to symbolically reconstruct a small story that we provided. We recruited three graduate students to encode the same story, including two from the sciences and one from our School of the Arts.

Corpus and World Model

In order to keep the task and its world knowledge relatively simple, we chose a story from the corpus of Aesop’s Fables titled “The Donkey and The Mule.” This fable includes a straightforward arrangement of characters, goals and actions:

A MULETEER set forth on a journey, driving before him an Donkey and a Mule, both well laden. The Donkey, as long as he traveled along the plain, carried his load with ease, but when he began to ascend the steep path of the mountain, felt his load to be more than he could bear. He entreated his companion to relieve him of a small portion, that he might carry home the rest; but the Mule paid no attention to the request. The Donkey shortly afterwards fell down dead under his burden. Not knowing what else to do in so wild a region, the Muleteer placed upon the Mule the load carried by the Donkey in addition to his own, and at the top of all placed the hide of the Donkey, after he had skinned him. The Mule, groaning beneath his heavy burden, said to himself: “I am treated according to my desserts. If I had only been willing to assist the Donkey a little in his need, I should not now be bearing, together with his burden, himself as well.”

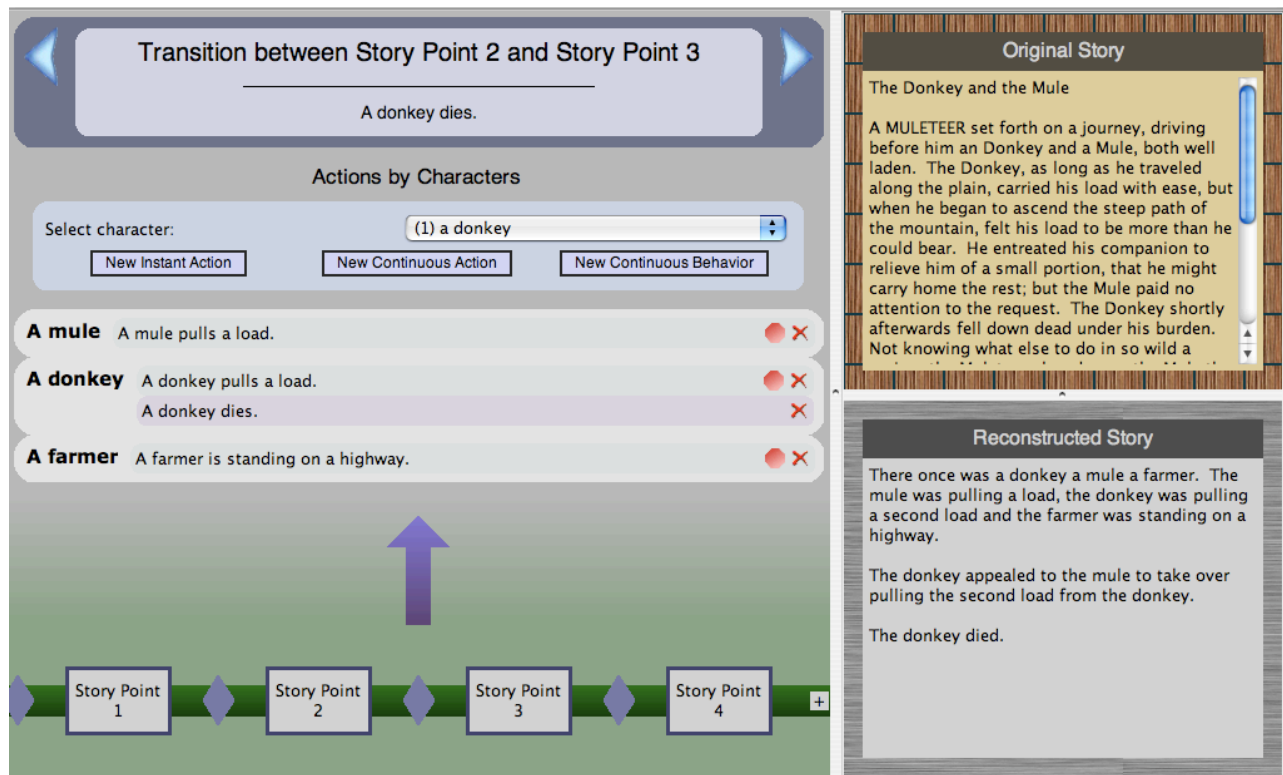


Figure 3: Screen capture of the sample tool's encoding interface that enables lay users to symbolically encode stories. For purposes of usability, conditions were named "behaviors," and states "story points."

The subjects were instructed to reproduce the story as closely as possible within the encoding interface. Additionally, they were told to model *implied* information not stated in the original story, in addition to explicit and pertinent facts.

To simplify the task, and enforce a constrained vocabulary so that results could be compared, subjects were not permitted to create new types of any kind of symbol. For the same reason, the interface simplified portions of the underlying model. For instance, as mentioned above, it allowed the subjects to only create goals, plans, and beliefs using particular types of actions and conditions (as opposed to diegetic timelines). Additionally, subjects were only allowed to access those transitions that linked adjacent states on the timeline. Figure 3 shows the encoding interface as configured in this manner.

The subjects were given a world knowledge model that included eleven condition types, nine action types, four characters among three character types, seven props among four prop types, and three locations among three location types. Some of the elements of the world model were designed to dovetail with the story at hand (e.g., the action *to die*); others were clearly irrelevant (e.g., *a prop explodes*). These facets were few enough in number that we could cover all the world knowledge with simple dropdown menus.

After subjects completed the task, they were asked three questions:

- Did you understand the task? If not, what was unclear?
- How easy or difficult was the task?
- Were you able to express everything you wanted to express using the menus? If not, what specifically did you find missing?

Results and Discussion

Subjects took between 15 and 30 minutes to complete the encoding task, in addition to 10 minutes of training. The following is a representative sample of the symbolically reconstructed story, as told by the generation component:

There once was a farmer, a donkey and a mule. The donkey was pulling a load, the mule was pulling a second load, the donkey was standing on a street, the farmer was desiring for the mule to pull the second load, the mule was standing on the street, and the farmer was desiring for the donkey to pull the first load. For a moment, the donkey wanted to pull the first load. The mule began to stand on a highway, the donkey began to stand on the highway, the donkey began to want to not pull the first load and the farmer began to stand on the highway.

The donkey appealed to the mule to take over pulling the first load from the donkey.

For a moment, the mule didn't want to pull the first load.

The donkey died.

The farmer forced the mule to pull the first load.

The mule told itself that it wants to begin pulling the first load.

The mule told itself that it didn't want to take over pulling the first load from the donkey.

The other reconstructions used similar subsets of world knowledge, though some included more interpretative details than others. When asked about the penultimate sentence, this particular subject reported that it was the closest he could come to modeling the donkey's expression that, as he paraphrased, "well, looks like now I have to carry the first load anyway."

In general, all subjects reported that the task was clear. Several described it as "fun." However, the subjects reported some difficulties constructing their models, which fall into three areas:

- **Missing narrative semantics.** Several subjects expressed a desire to model goals or beliefs involving past states (i.e., regrets about what should have happened) – a capability of the current model which the interface had eliminated for simplicity. The subjects also expressed a desire to model descriptive qualities of the objects in the story, such as the fact that the loads were heavy or that the mountain was steep. While these qualities could have been individually declared and invoked as conditions, a better adjustment will be to add support for a dedicated hierarchy of attributes.
- **Insufficient world knowledge.** Subjects mentioned some specific details of the original story they were unable to model due to missing world knowledge, such as the fact that the human was a muleteer as opposed to a farmer. However, the subjects did not report difficulty in finding the most appropriate symbols from among those available.
- **Interface implementation.** The majority of the subjects' feedback involved particulars of the encoding interface. For instance, they desired an "undo" function, a bird's-eye view of each state that includes thumbnail summaries for each character, and the ability to quickly locate assertions they had made by clicking on the corresponding fragment of reconstructed prose.

We are addressing these issues in current development. Of particular interest is the issue of forcing users to conform to a "controlled vocabulary" of world knowledge, as opposed to letting them "write in" new types of symbols. As we mentioned earlier, this type of restriction increases the likelihood that two different users will describe the same story element with the same symbol, and this agreement enables the system to compare and contrast their story graphs. On the other hand, static menus limit the expressive range of the tool, which may impact user satisfaction. Understanding the shape of this tradeoff, as well as the capacity of users to effectively design their own knowledge frames, is left for future work.

Future Work

There are several orthogonal directions for further work on SCHEHERAZADE. One is to refine and improve the underlying model. For instance, we will explore the benefits and complexity issues of adding further interpretative connections such as causality (explicitly linking an action to the condition that is its outcome). We will also look into conducting a more comprehensive study of the process of eliciting symbolic encodings from users.

The other direction is to build on SCHEHERAZADE and apply it to future experiments. We plan to create a more comprehensive world knowledge model for the domain of Aesop's Fables, sufficient to encode more than twenty different stories, and then elicit symbolic encodings from lay users. We plan to mine this corpus for thematic features that will enable statistical analysis using standard machine learning algorithms.

The modular nature of this representation allows for different flavors of analysis. For example, we can study the similarities and differences between the story graphs of a single story, varying the user dimension (to measure interpretative differences among populations) or the time dimension (to measure changes in recall across time). If subjects further annotate their story graphs with metadata labels, the learning algorithms could build models that predict these labels for *a priori* unknown graphs, with narrative features supplementing lexical ones. For instance, given a training set of graphs labeled as having happy or sad endings, the algorithms could predict whether other graphs have happy or sad endings. Moreover, they could discover the narrative trends of a genre by discovering the least-abstract patterns that hold across all the stories in the genre (as did Propp). In each case, we will study both the static representation of the *fabula* and the sequential nature of the *sjuzhet* used to tell the story.

The applicability of this approach for automatic story generation is unclear, but we believe that the narrative insights one gains from corpus analysis would be beneficial to prune the large search space that generation algorithms face.

Conclusions

We have presented SCHEHERAZADE as a foundational platform for formally representing, storing, and statistically analyzing stories, as well as the manner of their tellings. We have emphasized the versatility of the model, which can robustly handle stories of varying complexity and abstraction. We believe the platform can address the representational needs for a class of problems in narrative intelligence that includes understanding, co-authoring and narratology.

The key design decision to reach this aim is to separate out *narrative semantics* from both *world knowledge* and instantiated *story content*. The former is tightly managed, but sufficiently versatile to represent a range of human narratives. The narrative model we have encoded is an extension of established theoretical work on the structural morphology of narratives. Certain features of our representation, such as diegetic timelines, are novel compared to prior computational models of narrative. World knowledge, meanwhile, is

customizable rather than imposed *a priori*, so that higher-level tools can model their narrative domains to the depth and breadth that is appropriate on a per-task basis. Although SCHEHERAZADE cannot *solve* a narrative, as can a planning representation, it can detect thematic patterns suitable for statistical learning.

We also present a sample higher-level tool that allows users unfamiliar with narratology to symbolically encode stories with a graphical interface, and see the result rendered in prose with template-based natural language generation. When applied to the task of collecting symbolic encodings of existing stories from users (and, by extension, collecting their personal interpretations of those stories), this amounts to a new style of semantic annotation. Our formative evaluation of this process reveals that a small sample of users can create symbolic equivalents of traditional stories without great difficulty, and enjoy the process.

Acknowledgments

We thank David Plante, Rebecca Passonneau and our formative evaluation subjects for their valuable feedback.

References

- Alm, C., and Sproat, R. 2005. Emotional sequencing and development in fairy tales. In *Proceedings of the First International Conference on Affective Computing and Intelligent Interaction (ACII '05)*.
- Bal, M. 1997. *Narratology: Introduction to the Theory of Narrative*. Toronto: University of Toronto Press, second edition.
- Bruner, J. 1991. The narrative construction of reality. *Critical Inquiry* 18:1–21.
- Cassell, J. 2004. Towards a model of technology and literary development: Story listening systems. *Applied Developmental Psychology* 25:75–105.
- Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.
- Gerrig, R., and Egidi, G. 2003. *Cognitive Psychological Foundations of Narrative Experiences*. Stanford, CA: Center for the Study of Language and Information.
- Graesser, A.; Lang, K.; and Roberts, R. 1991. Question answering in the context of stories. *Journal of Experimental Psychology: General* 120:254–277.
- Halpin, H.; Moore, J. D.; and Robertson, J. 2004. Automatic analysis of plot for story rewriting. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '04)*.
- Labov, W., and Waletzky, J. 1967. Narrative analysis: Oral versions of personal experience. In Helm, J., ed., *Essays on the Verbal and Visual Arts*. University of Washington Press. 12–44.
- Mateas, M., and Sengers, P. 1999. Narrative intelligence: An introduction to the ni symposium. Technical Report FS-99-01. Menlo Park, California: American Association for Artificial Intelligence.
- Matuszek, C.; Cabral, J.; Witbrock, M.; and DeOliveira, J. 2006. An introduction to the syntax and content of cyc. In *Proceedings of the 2006 AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*.
- McKoon, G., and Ratcliff, R. 1992. Inference during reading. *Psychological Review* 99(3):440–466.
- Mueller, E. 2003. Story understanding through multi-representation model construction. In Hirst, G., and Nirenburg, S., eds., *Text Meaning: Proceedings of the HLT-NAACL 2003 Workshop*, 46–53. East Stroudsburg, PA: Association for Computational Linguistics.
- Passonneau, R.; Goodkind, A.; and Levy, E. 2007. Annotation of children's oral narrations: Modeling emergent narrative skills for computational applications. In *Proceedings of the 20th Annual Meeting of the Florida Artificial Intelligence Research Society (FLAIRS-20)*.
- Propp, V. 1969. *Morphology of the Folk Tale*. University of Texas Press, second edition.
- Riedl, M., and Young, R. M. 2005. Story planning as exploratory creativity: Techniques for expanding the narrative search space. In *Proceedings of the 2005 IJCAI Workshop on Computational Creativity*.
- Ryan, M.-L. 1991. *Possible Worlds, Artificial Intelligence and Narrative Theory*. Bloomington: Indiana University Press.
- Schank, R., and Riesbeck, C. 1981. *Inside Computer Understanding: Five Programs Plus Miniatures*. Hillsdale, NJ: Lawrence Erlbaum Associates.