A Rules-Based System for Adapting and Transforming Existing Narratives

Jo Mazeika(✉)

UC Santa Cruz, Santa Cruz, USA
jmazeika@ucsc.edu

Abstract. This paper describes a rules-based computational system that utilizes a semantic framework to produce transformations of an existing narrative. We describe how we can use a Rete rules system to transform a semantic representation of a narrative, as well as laying out groundwork for the types of rules that a system like this would consider. To provide an example of our system in action, we describe a semantic encoding of the Brothers Grimm version of Sleeping Beauty, and provide rules for transforming it to fit the style of Disney.

Keywords: Intelligent narrative technologies · Rules system · Narrative transformation · Narrative representation

1 Introduction

Narrative transformation is the act of modifying a pre-existing narrative, whether to serve some purpose or to fit an externally imposed constraint. We see a ready example of this in adaptations, where an existing narrative is taken and transformed to fit the conventions, strengths and limitations of a new medium. When books are adapted into films, for instance, characters are combined or removed and plot lines are dropped to allow the narrative to fit within the appropriate length of time. Additionally, when telling a story, we often change details to fit our audience. For instance, a parent will skip events deemed inappropriate for their children when reading a book, or the creators of a film adaptation of an old cartoon will make a new narrative to fit what they believe the now-older audience would like.

In this work, we propose a system designed to make targeted modifications to a semantic representation of a narrative, designed around changing the narrative to fit some higher level goal. We provide a rules-based system for describing the constraints and transformations that will be imposed on the narrative to fit the given requirements.

2 Related Work

Several different techniques have been commonly used for narrative generation. In [6], the authors provide four separate methods for generalizing Propp's theory

DOI: 10.1007/978-3-319-48279-8_16

of narrative grammars. The initial grammar that the authors describe operates over 5 levels of increasing specificity – the system first determines the high-level structure of the narrative, and at the bottom level, determines the individual actions that the characters take.

Additionally, we have systems such as Minstrel [13] and an improvement, Skald [11], which generate novel narratives. Minstrel operates by modeling some aspects of creativity to generate novel stories. To do so, the authors rely on transform-recall-adapt methods, or TRAMs, heuristics that offer solutions to problems that may arise during the generation process. In this way, it is able to transform the representations of narratives to fit an encountered narrative model. Using this technique, dubbed imaginative recall by Turner, Minstrel is able to generate novel narratives from an existing repository of reference narratives. Skald does not differ from Minstrel on the surface, but instead serves as a more-modern re-implementation that codifies and strengthens Turner's work by improving the TRAM selection process and updating several other mechanisms within Minstrel.

Finally, we have two systems that construct new stories from existing ones by analogy: the Story Translator [8], and SAM [14]. Both systems operate using very similar approaches, to the point of the authors in [14] observing that the Story Translator can be modeled within SAM. Both systems focus on taking existing narratives and converting them into an analogous one in a new domain; one example given is a fantasy story of a kidnapped princess being reworked to invoked cowboys holding each other at gunpoint. In contrast to the system described here, the authors view the changes in the events as a side-effect of the systems not being able to find a perfect mapping between the domains. However, these systems to provide an example of modifying a narrative to fit a given set of constraints.

3 System Description

Within our system, we have three representational elements that we need to consider: first, our representation of the narrative itself; second, the ontology that the rules operate within; and third, the set of rules that define the restrictions and transformations that we wish to impose upon the narrative.

To encode the stories that we wish to transform, we use a novel semantic framework, Rensa [5], to encode the representation of the narrative itself. Rensa is designed to be a generalization of Elson's Story Intention Graph [3], and as such offers all of the affordances that SIGs do while also incorporating a semantic network representation of the space around the narrative. In this framework, we represent characters as collections of mutable attributes, ranging from character tropes (protagonist, villain, etc.) to emotional attributes (happy, afraid, etc.). Because of the flexibility of Rensa, he particular set of attributes that comprise a character is defined by the user, and the set we choose is elaborated on in Sect. 5. The narrative itself is represented as a series of actions taken by the characters.

On top of the description of the narrative, we also provide the system with an ontology of the domain that the story takes place in, specifically, information about particular actions, character types Similarly to other works that use similar ontologies to generate stories [7,8,14], by defining this representation of the domain, we gain two major benefits. First, it allows us to reason over properties and actions that are not part of the initial narrative. If we want to substitute an action for a less violent one, for instance, we need a set of possible actions to choose from. By having and maintaining an action ontology, we are able to ensure that the action we choose fits within the setting for the narrative. Secondly, this ontology allows us to specify additional information about the encoded elements. For example, in order to remove violent actions from a narrative, we need some notion of how violent an action is. By including this non-narrative-specific information, we gain the ability to reason over the additional information that we have included in our ontology.

4 Rules Representation

With the narrative and the space of the narrative we wish to generate in, we now need to build our series of rules for transforming the narrative. To do this, we take our semantic representation and encode it in a Rete algorithm-based rules system [4], which allows us to efficiently identify and execute rules over a collection of statements. The state is represented as a collection of facts, which in our case are simply the assertions that we have created to represent our narrative and the information contained within the ontology.

Rules can have three different effects when they fire: they can add any number of new facts to the system, they can modify any number of existing facts, or they can remove existing facts. When the state is changed in this way, we have might have a new space of rules that will be able to fire next.

In this system, we implement several major classes of rules that represent different kinds of transformations that we want to be able to make to the narrative representation. Note that not all of these can be represented as a single rule, but sometimes instead must be implemented as a set or series of interrelated rules to produce the effect desired. Side effects are certainly possible, but should be minimized for rules that may need to fire multiple times.

First, we have a collection of general types of rules that we use within the system. These include rules to include additional information or actions as required (for instance, some narratives will require all of their characters to have names), to modify or remove existing components of the narrative, and, most importantly, to ensure consistency within the narrative because of these modifications. As the rules fire, we may find ourselves in states that are not valid – perhaps an action is added to the narrative, but the character performing the action does not exist at that location at that time. The consistency rules fire to trigger modifications to ensure that these issues are resolved.

With the general sorts of rules in place, we then incorporate a rule set that defines the requirements for the target narrative. Beyond rules that can be modeled in the above class (such as the removal of violent actions), we can include rules such as requiring a character of a particular archetype to be present (whether that character has to be added to the narrative on the fly, or an existing character can be modified to fit the archetype).

Once we have our rules set in place, constructing the modified narrative happens by allowing the Rete algorithm to use the rules to modify the narrative until it reaches a state in which no further rules can fire. At that point, we take the current state of the engine and transform it back into the semantic representation, so that we can examine the output.

5 Example

To test this framework, we focused our efforts on constructing an adaptation the story of Sleeping Beauty. Since there's a culturally salient transformation for that particular story – namely the Disney adaptation of the original story from the Brothers Grimm. To do this, we used the text provided at http://www.pitt.edu/~dash/grimm050.html as our reference for the original tale [2].

First, we discuss how we have encoded the narrative into the system. Characters are initialized with the following attributes: Name, Gender, the character's Narrative Role (Protagonist, Villain, Sidekick, etc.), and any other necessary attributes (Character is a faerie or an animal, etc.)

Then, the narrative is comprised of three different kinds of statements: a character can take an action, a character can move to a new location, a character can acquire a prop, or a character can gain or lose a property. Each of these has the following attributes:

- Actor (The focus character)
- Type (Action, Location, Prop, Property)
- Subject (The Action taken, the Location moved to, the Prop acquired, or the property gained or lost)
- Time Step(s) (This statements can have one of two time step attributes – either the statement happens at a single time step, or it has a time step where it starts and a time step where it finishes)

In our encoding of Sleeping Beauty, the narrative happens over 12 time steps, consisting of 26 different statements.

The system uses the following rules to create a mapping from Grimm to Disney:

- Every character has a name.
- The hero has an animal companion as a sidekick.
- Repeated characters, or characters who serve identical narrative functions, are keep to a minimum of two.
- The villain is defeated.
- No excessively violent actions occur.

These rules are not entirely inclusive of the rules that would transform the narrative to match the Disney version of Sleeping Beauty – for instance, there is no rule that handles Maleficent transforming into a dragon in Disney version.

Additionally, our encoding is on a high enough level that some features aren't considered, such as the individual blessings that the different fairies grant to Sleeping Beauty. However, the point of these rules is not to ensure that the actual Disney script is re-created, but instead are focused on transforming the narrative into one that fits within the style of the Disney.

Figure 1 shows an example of the expression of the first rule. In the original narrative, many of the characters are not actually given names: the king and queen, for instance, are only referred to by their titles. To demonstrate how we correct this, we take a character (in this case, Sleeping Beauty) who doesn't have a name. Names are a property of characters, and so to give her a name, we need to create a new fact that says what her name is.

We implement this by having a rule that fires when there is a character who doesn't have a name property. When said rule fires, we pull randomly from a gendered list of names, and create a new fact within the system that says that character has the chosen name. If the character does not have a gender at this time, we resolve this by randomly assigning a gender and then a corresponding name. In Fig. 1, we see that the character "Sleeping Beauty" has the name Aurora chosen and assigned.

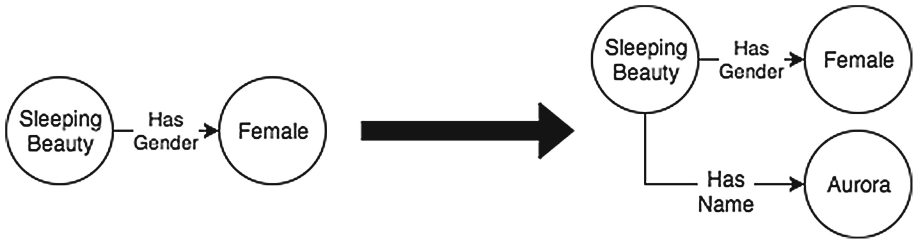

Fig. 1. A realization of the first rule.

The animal sidekick rule is similar. If any animal already exists within the narrative, we give that animal the property of sidekick; otherwise we create a new animal with the property in place from the get-go. Afterwards, we then include what being a sidekick means. In this case, the property we wish to include is that at each time stamp, the sidekick and protagonist are both in the same location.

Next, we have the repetition reduction. This isn't necessarily an important function – plenty of narratives can feature a large number of characters who all take the same actions. This particular rule was created with the Good Fairies from Disney's Sleeping Beauty in mind. The original Grimm tale instead featured twelve wise woman who filled the same role.

In this, first we need to get a count of the number of characters who all share the same set of actions over the course of the story – specifically, characters who all do the same thing at the same time. When we have three or more characters who fall under that criteria, we select one of them for removal – since all of these characters are considered identically under the narrative, it does not matter

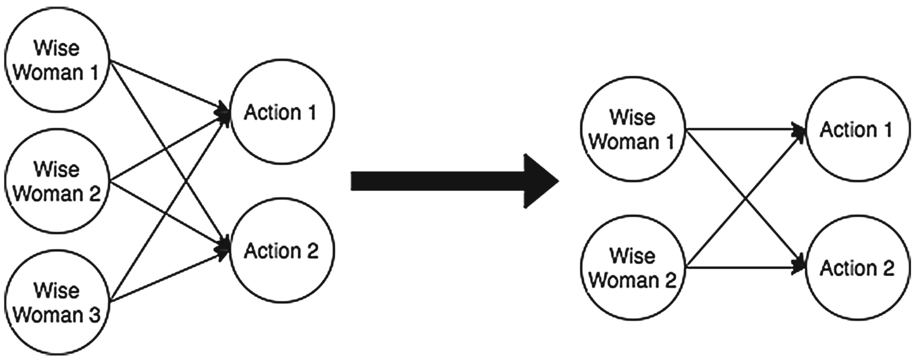

Fig. 2. Removing an extraneous character.

which one we select for removal. We then need to remove all of the lingering associations and facts that involve that the removed character. This is shown in Fig. 2.

For the next rule, we ensure that the villain is defeated. This is requires multiple different checks – if an action in the narrative itself would cause the villain's defeat, then it is a simple matter of appending that particular piece of information. Otherwise, we need to find an action that causes defeat, and have it performed. Most of these actions are performed by a different character (although there is no reason that the villain couldn't perform the action that causes their defeat), and so we need to ensure that character is in the correct place in the story to perform that action. Finally, if the action contains any prerequisites, we need to ripple back through the story to ensure that all of them are met before continuing.

In Fig. 3 we model a transforming part of the story to remove an action that has been labeled within the ontology as too violent. This notion of what qualifies as too violent is subjective, and several Disney movies do feature their villains coming to rather bad ends – in particular, Disney's Sleeping Beauty features Maleficent being stabbed with a magical sword. However, for the purposes of illustration, we have picked that particular scene as too violent, and in need of replacement.

To do this, we first need to find a replacement action that has the same set of consequences – in this case, we're looking for a non-violent action that causes defeat for Maleficent. Our system identifies "Break Weapon" as a replacement for "Stab", and so substitutes that in its place. However, both of the actions have preconditions: while Stab requires a blade, Break Weapon requires that the character has the property Distracting. So, we give the character the Distracting property. If the character has no other actions that require them to have a blade, we can also remove that from the character's list of properties as a tidying measure.

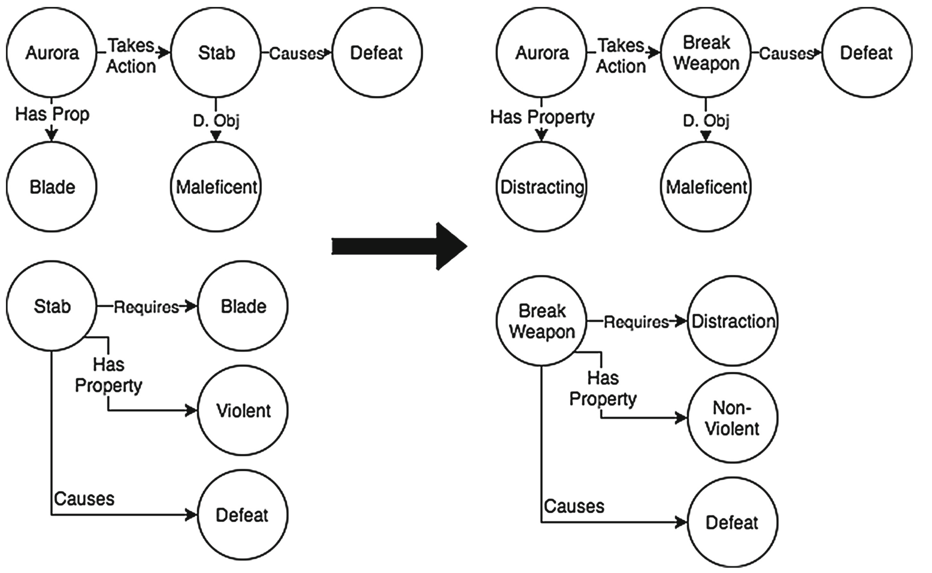

Fig. 3. Replacing a violent action.

6 Discussion and Future Work

While this work has shown reasonable results over the space that its generated in, it is not without limitations. The most obvious is that the Rete rules system used within this system is deterministic; if the same set of conditions and rules are true at any point during execution, the same set of rules with always fire in the same order. While this is useful under certain circumstances, for this work it is a limitation. We want to be able to explore the space of possible stories, which we can do by choosing different possible orders for firing rules, or by choosing different facts to match with the preconditions.

Next, the ontology used for generation is hand-authored. While is it certainly possible to have a system learn the ontology of fairy tales (both Disney and Grimm) and use that for generation, we chose for this initial outing to focus on a small, handcrafted ontology.

Finally, the pipeline that this sort of generation would be a part of is not complete. While we are able to manipulate the narrative on the representational level, we do not yet have a mapping from the story representation to natural language.

However, this work does lay the groundwork for the constructing narratives that fit a particular style, as well as an example of doing so. Future research will allow us to refine and improve the both the particular rule sets and the generation system.

References

1. Akimoto, T., Ogata, T.: A narratological approach for narrative discourse: implementation and evaluation of the system based on genette and jauss. In: Proceedings of the 34th Annual Conference of the Cognitive Science Society, pp. 1272–1277 (2012)
2. Ashliman, D.L.: Grimm Brothers' Home Page. Accessed 2 June 2016. http://www.pitt.edu/dash/grimm.html
3. Elson, D.K.: Detecting story analogies from annotations of time, action and agency. In: Proceedings of the LREC 2012 Workshop on Computational Models of Narrative, Istanbul, Turkey (2012)
4. Forgy, C.L.: Rete: a fast algorithm for the many pattern/many object pattern match problem. Artif. Intell. **19**(1), 17–37 (1982)
5. Harmon, S.: An expressive dilemma generation model for players and artificial agents. In: Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference (2016)
6. Imabuchi, S., Ogata, T.: Methods for generalizing the propp-based story generation mechanism. In: Yoshida, T., Kou, G., Skowron, A., Cao, J., Hacid, H., Zhong, N. (eds.) AMT 2013. LNCS, vol. 8210, pp. 333–344. Springer, Heidelberg (2013). doi:10.1007/978-3-319-02750-0_36
7. Ogata, T.: Building conceptual dictionaries for an integrated narrative generation system. J. Robot. Netw. Artif. Life **1**(4), 270–284 (2015)
8. Riedl, M., Len C.: Generating story analogues. In: AIIDE (2009)
9. Rishes, E., Lukin, S.M., Elson, D.K., Walker, M.A.: Generating different story tellings from semantic representations of narrative. In: Koenitz, H., Sezen, T.I., Ferri, G., Haahr, M., Sezen, D., Çatak, G. (eds.) ICIDS 2013. LNCS, vol. 8230, pp. 192–204. Springer, Heidelberg (2013). doi:10.1007/978-3-319-02756-2_24
10. Speer, R., Havasi, C.: ConceptNet 5: a large semantic network for relational knowledge. In: Gurevych, I., Kim, J. (eds.) The People's Web Meets NLP. Theory and Applications of Natural Language Processing, pp. 161–176. Springer, Heidelberg (2013)
11. Tearse, B.R., Mawhorter, P.A., Mateas, M., Wardrip-Fruin, N.: Skald: minstrel reconstructed. IEEE Trans. Comput. Intell. AI Game **6**, 156–165 (2014)
12. Theune, M., Slabbers, N., Hielkema, F.: The automatic generation of narratives. In: Proceedings of the 17th Meeting of Computational Linguistics in the Netherlands (CLIN17), pp. 131–146. Leuven, Belgium (2007)
13. Turner, S.: Minstrel: a computer model of creativity and storytelling. Technical report CSD-920057, Ph.D. thesis, Computer Science Department, University of California, Los Angeles, CA (1992)
14. Zhu, J., Ontañón, S.: The sam algorithm for analogy-based story generation. In: Artificial Intelligence, pp. 67–72 (2011)