The Thirty-First International Florida
Artificial Intelligence Research Society Conference (FLAIRS-31)

Let Us Tell You a *fAIble*: Content Generation through Graph-Based Cognition

Vera A. Kazakova,[*] Lauren Hastings,[*] Andres Posadas,[*] Lucas C. Gonzalez,[*] Rainer Knauf,[**] Klaus P. Jantke,[***] Avelino J. Gonzalez[*]
[*]Department of Computer Science, Univ. of Central Florida Orlando, Florida
[kazakova@cs.ucf.edu, hastingsl@knights.ucf.edu, andres.posadas@knights.ucf.edu, lcgonzalez@knights.ucf.edu, gonzalez@ucf.edu]
[**]Technische Universität Ilmenau Ilmenau, Germany [Rainer.Knauf@tu-ilmenau.de]
[***]ADICOM Software GmbH Weimar, Germany [klaus.p.jantke@adicom-group.de]

Abstract

In this work we present *fAIble*: a novel graph-based modular storytelling framework. *fAIble* is centered around a graph database, incorporates invariable elements of folktale structure, while accounting for thoughts and actions. Action outcomes are a product of probabilistic story generation. Probabilities are based on elements of common sense, invariable elements of folktale structure, high-level character roles, and a wide variety of other variables (e.g. characters' physical and psychological traits, context-based likelihood of encountering specific items and characters, etc.). A prototype implementation is tested through an anonymous questionnaire. Results demonstrate the ability of graph-based cognition to produce coherent story prototypes with sensible character actions, while maintaining output variability.

1 Introduction

In this work we present a novel graph-based narrative generation system, *fAIble*, developed to be highly scalable through ensuring that new features and narrative elements can be incorporated at minimal cost. *fAIble* is also highly extensible, allowing for new narrative generation components to be easily weaved into the story creation process. In our preliminary implementation, *fAIble* aims to produce variable yet coherent stories composed of sensible character actions.

In automated narrative generation, a set of story events are generated with some partial ordering and then transformed into sentences describing these events. Narratives can be generated from scratch or from predefined elements, such as readers' preferences. Our initial focus for *fAIble* is on automated bedtime story generation for children, as a reasonable first step toward generating narratives suitable for more complex purposes. Teachers often try to come up with interesting and memorable analogies and stories to explain complex concepts. Archaeologists often combine their findings with educated guesswork to produce potential narratives of the past. Business analysts look at the data and attempt to make out the underlying market dynamics. In Massively Multiplayer Online Role-Playing Games, many characters with interesting and interconnecting backstories are often needed to keep players engaged. Automated narrative generation can facilitate development in all of these fields.

While bedtime stories are often not as complex as other narratives, they have their own unique requirements, such as seemingly endless variability for any session preferences and an ability to quickly and robustly adapt to mid-session alterations from a reader. Path-planning is commonly used in narrative generation to ensure story viability from start to finish (e.g. MINSTREL performs planning through repeated graph searches (Turner 1993)), decreasing system scalability (as searching is slow in large spaces) and hindering content variability (as invalid paths must be discarded) . *fAIble* eliminates path-planning as the main driver behind event generation to create fast, robustly reactive, and variable narratives. *fAIble* ensures that steps can be taken in any situationally feasible direction, with the system introducing events or elements to circumvent any optionless situations, allowing narratives to be generated one reactive step at a time.

Some notable examples of narrative generation that do not search for complete paths from start to to finish are MAKEBELIEVE (Liu and Singh 2002) and CAMPFIRE (Hollister and Gonzalez 2017). MAKEBELIEVE requests a starting sentence from the reader and follows free-form association to arrive at the next event, and then the next, and so on. With no goal in mind, there is no need for planning, but the associations require a commonsense database. In *fAIble*, stories do have end goals, requiring a more structured approach, for which we also employ commonsense knowledge. While the commonsense facts and connections used in the presented prototype are manually authored, in the future, *fAIble* could benefit from incorporating knowledge contained within the ConceptNet reasoning tool-kit (Liu and Singh 2004). CAMPFIRE avoids conventional path-planning through use of templates, filling in elements with any one of several equivalent options. While *fAIble* also employs templates, these are more general, and probabilistically interweaved with each other during story generation, drastically increasing overall plot-line variability.

This work is part of ongoing research to develop avatar-based adaptive storytelling for children and continues the work started in the AESOP system (Wade et al. 2017). AESOP begins by creating a StorySpace containing all combinations of actions, characters, and objects. These are subsequently distilled down to a subset based on some initial story conditions. Scaling such an approach is prohibitively expensive. Both *fAIble* and AESOP focus on generating a *fabula* (a chronological sequence of events) in an effort to build a solid foundation for future development of *syuzhet* (different orderings of the events as means of artistic expression, building suspense, etc.) (Shklovsky 1991). Thus, stories are currently presented in their linear order. Inspired by CAMPFIRE and AESOP, characters' actions and their outcomes are based on characters' traits, skills, and relationships, which change as a result of their interactions. Nevertheless, AESOP allows character actions extraneous to the plot. *fAIble*'s characters employ common sense contextual reasoning and thus *fAIble*'s fabulas only contain events directly relevant to the story and the characters' goals.

This project is composed of the following three phases: (I) creation of an automated storytelling system, (II) incorporation of interactive as well as reader-profile-based story modification elements, and (III) use of conversational avatars to relate the story to a child. In this work we focus on developing the infrastructure necessary for content generation of part I, in a way that will make the system easily extensible to handle user input of part II, and thus will make the system amenable to interactive avatar narration in part III.

This work is subdivided into two parts aimed at creating the *fAIble* automated narrative generation system: (A) development of extensible modular setup of interconnected story generating elements and (B) processing of auto-generated story events into English sentences intended to improve how the story sounds as well as convey meaninfgul additional information through descriptors. First we describe the aims of the *fAIble* system and how these aims inform the design and interaction of the system's modules. We then describe how *fAIble*'s Natural Language Generation (NLG) transforms story events into English sentences. Subsequently we provide an overview of how a *fAIble* story is generated, followed by a report on preliminary system testing and results.

2 *fAIble*'s Templates, Modularity, and Graph-based Cognition

The *fAIble* storytelling framework focuses on employing maximally general and highly interconnected templates that provide the system with commonsense information, resulting in minimally restricted yet sensible outputs. Since sensible stories are directly related to sensible character behavior, we employ context-based reasoning and incorporate transparent character thinking, providing readers with insights into the minds of the characters. *fAIble* also focuses on improving scalability and online performance, while facilitating future accommodation of requests for changes to the autogenerated narratives. To this end, *fAIble* eliminates the need for path-planning by ensuring there are no narrative dead ends. In this section we discuss how *fAIble* addresses modularity, extensibility, and variability.

Templates of Common Sense Decreasing the authorial burden while generating sensible and variable narratives can be difficult. The use of narrative templates is generally undesirable, but it is cost-effective given the resulting gains in narrative quality. Templates do, however, represent a constraint on system creativity, so their use should be minimized. Thus, the question becomes how to best set up the infrastructure of the narrative space so as to minimize required input while maximizing output quality and variability.

Humans decide whether stories make sense based on either pre-exisiting or story-supplied information. World rules allow a story to make sense, even when the world is not our own. Characters are expected to be motivated by their circumstances, experiences, goals, capabilities, and personalities. Children are taught aspects of world dynamics over the course of growing up, progressively developing their common sense through acquisition of common knowledge. In order to create coherent narratives composed of sensible character actions, the system must also have a way of knowing how the story world works. This type of "AI upbringing" can be costly, given the need for system designers to supply all relevant knowledge to the narrative generation. Nevertheless, the aspects that lead readers to decide whether a narrative makes sense persist across stories. Thus, abstracted commonsense reasoning and knowledge templates constitute a comparatively small and potentially distributed investment that can be shared among narrative generation systems.

Such a large and complex infrastructure would be developed incrementally. Therefore, extensibility of the individual modules as well as of the system as a whole by means of new components is paramount. Implementing large scale character and narrative development has often been tackled through modularity and reuse. Even human authors often follow rules and patterns to produce narratives, conforming to a set of variable and invariable story elements. Propp generalizes Russian folktales' invariable elements into only seven character roles (e.g. Hero) and 31 narrative themes or narratemes (e.g. Hero's Return), while variable elements include character motivations, personalities, experiences, and goals (Propp 2010). Consequently, in *fAIble* we focus on a modular design with strongly interweaved components, with the intent to capture elements of common sense.

The *story space* is defined as the set of all the physical and abstract elements present in the narrative world (Kybartas and Bidarra 2017). Many potential sources of common sense can inform story space elements, as well as how they interact. Setting up every source as an individual reusable component to be queried by the system as needed affords maximal behavioral gain from each engineered element. To create new environments, a collection of prefabricated elements can be used, containing templates and options for variables such as character types, names, items, as well as different types of physical and psychological character traits, and character relationships (Lebowitz 1984; Pickett, Khosmood, and Fowler 2015; Wade et al. 2017).

In *fAIble*, templates are used to supply the system with common knowledge of story-world-specific character classes, which items and locations are most commonly seen given a character class, generic high-level quest lines, and contextual reasoning. Templates are also used for character relationships. As in AESOP, these relationships change over time, but in *fAIble*'s modular approach, a hierarchy between feelings and relationships is established. Relationship changes are triggered by interaction-based shifts in character feelings toward one another. This affords additional granularity and overall more sensible character behaviors. The

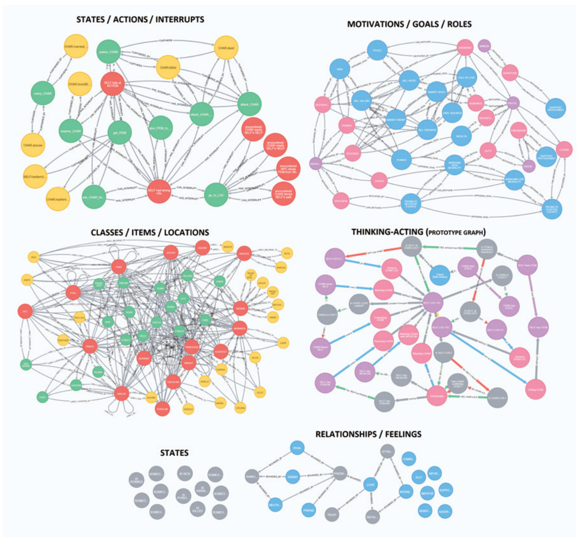
STATES / ACTIONS / INTERRUPTS
MOTIVATIONS / GOALS / ROLES
CLASSES / ITEMS / LOCATIONS
THINKING-ACTING (PROTOTYPE GRAPH)
RELATIONSHIPS / FEELINGS
STATES

Figure 1: Entire *fAIble* story space in Neo4j

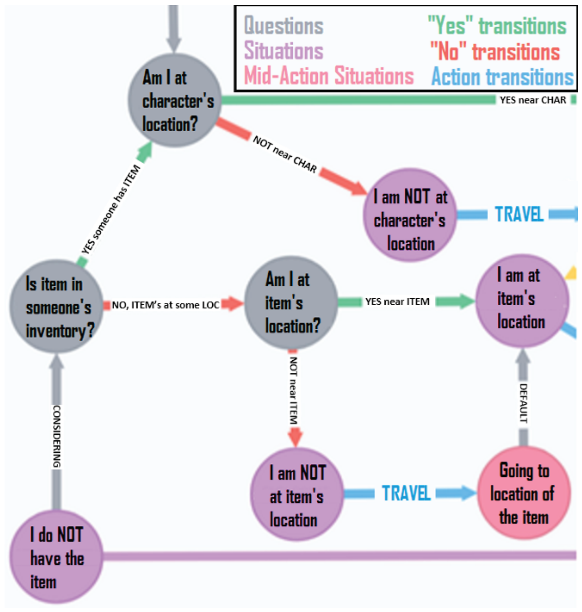

Figure 2: "Acting-Thinking" Graph in Neo4j

fAIble prototype story space is presented in fig.1. Throughout the system, templates are used probabilistically, allowing for a portion of story elements to defy expectations. In this way, we strive to maintain interest, while moving away from predictable and unnecessary physical, psychological, and other kinds of stereotypes. As a result, in one autogenerated story, a villainous princess robbed our hero thief, who then had to set out on a quest to recover the precious item.

In *fAIble*, we group small sets of ordered narrative elements into predefined high-level quest-lines. *Quests* affect story-level goals, and specify general character goals. Then, characters are left to decide how to achieve their goals. Quests allow the story to select a high-level quest type and specify a small set of skeletal events indicating potential starting points, trigger events, main goals, and potential outcomes. Such narrative landmarks have been previously used to address scalability issues in path-planning narrative generation (Porteous, Cavazza, and Charles 2010). In *fAIble*, however, their use is closer to Propp's narratemes (Propp 2010). Each of the substeps in our quest templates represent a landmark the story must reach before moving on to the next. For example, the "saving a poisoned loved one" quest is defined by the following events: (1) protagonist is assigned a secondary familiar character, (2) who becomes poisoned, and (3) the protagonist must obtain an antidote. Long term, even these high-level templates can be removed with the inclusion of additional world knowledge that would allow inferences such as that a poisoned character requires an antidote, or that needs of loved ones become our own.

Interweaved subplots are common in human-generated narratives and are an important driver behind story complexification and variability in *fAIble*. We allow actions to be interjected with action-type specific "Interrupts". In this way, additional quests can be introduced probabilistically mid-story, turning Quests into yet another reusable template within *fAIble*. Interrupts add variability and unpredictability by adding characters and subplots, making achieving goals more difficult, and injecting world events into the story (e.g. a character tries to cross a bridge, but the bridge collapses, leaving the character to find another way to the goal). Narrative duration is also controlled through these probabilistic interrupts, which in the prototype become less likely as the story progresses, allowing existing quest arcs to wrap up.

Context-Based Reasoning and Character Thoughts We employ high-level CxBR (Gonzalez, Stensrud, and Barrett 2008) to ensure knowledge is reusable throughout the system and immediate story choices are limited to a small subset of hierarchically arranged options. On every step, characters probabilistically select from among the available options, restricted by the encoded general notions of common sense. The system can allow for more commonsense conditions, such as additional specific world rules, to be incorporated without substantially increasing runtime or space complexity, since only contextually relevant options are considered, ultimately allowing for more complex decision-making. With CxBR, linear increases in the system's modules and options lead to exponential increases in variability.

fAIble's graph database houses a manually designed "Acting-Thinking" graph intended to represent an abstracted version of common sense that characters use to pursue their goals. This graph is designed to be high-level and applicable to abstracted general situations, such as "obtaining an item". A portion of this graph is shown in fig.2. Abstract Situation nodes in the Acting-Thinking graph are intended to allow characters to consider all high level contexts possible within the narrative (although the prototype only has a small subset). Nevertheless, the graph does not directly represent the entire space of possible states. Instead, it is a template in its own right, where every node only considers the presence or absence of a small set of commonsense facts, alongside character- and situation-specific values for the variables embedded within the node's considerations. *fAIble*'s Situation contexts are maximally abstract, structured as a simplified version of human thinking, allowing for relatively small graphs to represent large space of possibilities. All specifics are encoded as variables to be filled in with relevant story elements when a Situation is reached during narrative generation. An action is selected from the set of available actions based on the story space and the individual capabilities of the characters. Once at a Situation node, many actions may become available. Some do not need to be on the graph, as they are always available non-context-based actions. Other actions are Situation specific, like attacking. The active character then estimates which of the available actions is best suited for the immediate goal based on character specifics, situation specifics, and the most immediate subgoal (only one goal is considered at a time in the prototype), followed by a probabilistic choice from among the viable alternatives.

Character actions are a product of their "thinking", i.e. navigating the defined Acting-Thinking graph, which represents a partial ordering among highly abstracted contexts. Additional transitions altering a character's situation are also possible through story Events. To produce sensible characters, *fAIble* focuses on considering actions as a product of motivations and goals, as well as of the characters' psychological and physical attributes. To create seemingly reasonable behaviors, *fAIble* allows characters to ask themselves the types of questions a human might when considering courses of action. As a character considers what to do next, it goes through a series of contextual thoughts, such as "Is opponent Evil?", and the answers affect the resulting action choice. Subgraphs can be used to organize thinking hierarchically. This approach allows characters to arrive at sensible conclusions and provides the option of explicitly incorporating transparent thinking into the narrative. In the future, thoughts can be used to present complex emotions or characters whose actions appear evil while their thoughts are not.

Replacing Planning with Adaptability Children can be unpredictable. Skilled storytellers are flexible to initial requests and mid-story changes and often do not plan the entire story in advance, allowing for near instant startup and infinite adaptability to run-time changes. Interactive storytelling targeted for children should thus focus on responsiveness and ability to adapt to requests. To incorporate maximally open requests, the system will ultimately need large amounts of commonsense knowledge, exacerbating path-planning-based scalability issues. In *fAIble*, search is minimized by ensuring high adaptability to unexpected events.

Scalability issues stemming from path-planning are addressed by removing planning as the main narrative driver. Planning is commonly used to ensure viable paths between the start and end of a story. Instead of searching for such viabilities, we ensure that no story point is a dead end. If story events place a character in an impossible to resolve situation, Deus ex Machina events can be generated to alter the situation. Eliminating narrative dead ends makes the system more scalable to real world domains, as *fAIble* only needs to make a single narrative step at a time. Since only the following step must be feasible, more of the situations become reachable within a story, further increasing narrative variability.

Graph Structure as a Design Tool In this work we focus not only on a novel narrative generation system, but also on a novel way of designing the system itself through the use of the graph database Neo4j. Graph-based storage lends itself naturally to contextual reasoning: nodes in the "Thinking-Acting" portion of our database represent abstract contextual situations, interconnected via actions and events (see fig.2). The graph also provides a direct way to visualize system components and their interconnections (see fig.1). Complex interactions among elements within the story space can be difficult to design. Graph databases provide an intuitive way to design complex worlds and interactions, allowing us to improve narrative output by facilitating conceptual design.

3 *fAIble*'s NLG

Story events generated within *fAIble* require linguistic processing prior to being output as natural sounding and meaningful sentences. *fAIble*'s NLG is responsible for accepting the output of the Story Generation component consisting of a series of ordered event items. Each event contains elements that specify what was done, by whom, to whom, as well as any additional elements (e.g. with what item). During processing, these elements are transformed and enhanced, prior to being recombined into a format amenable to sentence formation by the SimpleNLG engine (Gatt and Reiter 2009).

The NLG module in our prototype focuses on enhancing language variability while forming sentences corresponding to story events. Variation is increased through redundancy reduction, character identifier variation, and descriptor-based enhancements. Event redundancy is reduced through basic event aggregation, which has been shown to improve narrative quality (Dalianis and Hovy 1996). Identical adjacent events are combined into a single multi-count event; e.g."The prince hit the creature three times." Language is further varied through character references alternating between their names and class types. Character action events are enhanced with character-specific adverbs. Action enhancements through adverbs have been used in (Barber and Kudenko 2009) to convey characters' dispositions toward events. In *fAIble*, adverbs are added to action events to give readers an insight into characters' physical attributes in an interesting and unobtrusive manner. For example, a character who "clumsily attacked" someone will give the reader am implicit sense of low dexterity. As characters' attribute values affect action outcomes, these descriptors also hint at events' results, without a need for unnatural and disruptive explicit updates. For example, the amount of damage caused by an attack executed "expertly", implicitly indicates a high health point loss by the opponent. In the future, implicit expectations can be used to describe surprising outcomes such as "Unbelievably, the monster remained unharmed."

To produce more natural sounding stories, *fAIble*'s NLG module keeps track of what the reader knows. New elements are introduced with indefinite articles, while subsequent mentions are accompanied by definite articles (e.g. "The princess encountered a witch. The witch asked for a cauldron."). Locations are introduced alongside their descriptive elements only once, upon first encounter. Likewise, character thoughts pertaining to identical story elements are only displayed the first time these thoughts are considered.

The NLG in *fAIble* relies on the SimpleNLG (Gatt and Reiter 2009) language generation engine as its backbone. SimpleNLG is a *realisation engine*: it focuses on the mechanical combination of sentence elements into a string, once the semantic inputs have been pre-assigned to the desired syntactic outputs. In this way, *fAIble* is responsible for specifying how elements of each event line up with syntactic elements of the generated sentence, through specifying syntactic attributes for all action nodes within the graph.

Note, however, that in the current prototype, character thoughts are fully formed as strings on the story-generation side and thus are not subject to further processing within the NLG module. In the future, thought-specific NLG processing will be implemented to allow for more complex thought-result combination outputs. Location adjectives currently are treated as randomized flavor-text. In the future, these descriptors will be associated with location events. For example, "perilous" locations will have more negative terrain-based events, such as path obstructions, while a "crowded" location will have more character encounters.

4 *fAIble* Story Generation Process

The *fAIble* narrative generation system is composed of three main components: 1) a graph database, 2) an event generating component, and 3) an NLG component. In this section we overview the different elements of *fAIble*, as well as how they interact to produce a story.

All elements and their connections are stored in the graph database *Neo4j*. Neo4j loads and interconnects all the related components from JSON files using its CYPHER querying language. JSON files are manually populated with story-elements such as character class types and attributes, quest types, location types, relationship types, feelings, actions, action interrupt options, items, as well as types of goals, motivations, and character role types. The main driver behind *fAIble* is written in Java 8. The program queries the database using CYPHER. Querying is first done to bring only the necessary elements from the database into local memory, i.e. only the elements selected for the story are loaded into the program. Subsequently, the "Acting-Thinking" graph is queried at every step, allowing characters to consider only what is relevant to their situations. Once a character action or a story event is generated, it is passed to the third major component, also written in Java: *fAIble*'s NLG. The individual elements are augmented by *fAIble*'s NLG before being passed into SimpleNLG for sentence formation and subsequently returned to be output as part of the story.

We now review the entire story generation process. First, *fAIble* randomly selects a single main quest type from those available in the database (e.g. "avenging the death of a loved one"). Then, the roles needed for that quest type are filled in from the available character classes. Character attributes are assigned stochastically, within acceptable value ranges for each class. Character relationships, motivations, companions, starting items, and initial locations are set probabilistically, but can be further informed by high-level quest

Our story begins in a village. A mercenary named Nathaniel lived in the village. The village was welcoming. Nathaniel had a companion named Christopher. Daria, the creature, killed Christopher. Nathaniel set out to avenge Christopher. Nathaniel thought: "Is Nathaniel at location of Daria: forest?", "No" he thought. Nathaniel went to a forest. The forest was sunny. Nathaniel thought: "Is Daria dead?", "No" he thought. Nathaniel thought: "Is Nathaniel evil?", "No" he thought. Nathaniel thought: "Is Daria evil?", "Yes" he thought. Nathaniel thought: "Nathaniel has weapon?", "Yes" he thought. Nathaniel attacked Daria violently with a sword. Nathaniel tried to attack Daria with a shield but failed. The creature escaped to a road. Nathaniel thought: "Is Nathaniel at location of Daria: road?", "No" he thought. Nathaniel went to the road. Nathaniel tried to attack Daria with the shield but failed. Nathaniel attacked Daria violently with the sword. Nathaniel killed the creature. Nathaniel thought: "Is Daria dead?", "Yes" he thought. The End.

Figure 3: A sample *fAIble* auto-generated story.

requirements. A final goal and an opposing starting condition are selected based on the quest type. The protagonist selects one of its available goals, ordered by urgency, and uses high-level filtering via the "thinking-acting" graph to decide on a single action, which is then sent to language processing. The outcome of this action depends on character attributes as well as on a probabilistic interrupting mechanism. Interrupts can result in action failures, character encounters, new mandatory or optional side quests of varying urgency, or world events. Action outcome is then also sent to language processing. Any new goals are added to the protagonist's goal stack in accordance with goal urgency and optionality. Mandatory side quests are those presented by elements standing in the way of completing the current quests, while optional side-quests have no direct relation to the quest at hand. The protagonist then continues "thinking-acting" until resolving the stack of active quests. A sample story is provided in fig. 3. Note that although multiple characters are present, in this prototype only the protagonist is afforded agency. All other characters are only involved in events inciting the protagonist to act. Thus, although failure is allowed, since antagonists are not currently afforded agency, the protagonist always fulfills its goals. Note also that in this prototype user preferences are excluded to ensure the system is fully functional with no user input. In future implementations, users will be allowed to alter any of the choices currently made by the system.

5 System Testing and Results

Narrative quality depends on many subjectively evaluated elements, but plot coherence and character believability have been outlined as uniquely important to successful narratives (Riedl and Young 2010). *Plot coherence* is the perception that story elements are meaningful and relevant to the story; *character believability* is the perception that characters' behaviors are a product of their beliefs and desires. (Riedl and Young 2004) Additionally, in the absence of visuals, language is solely responsible for relating the generated stories to the readers. For evaluating *fAIble*'s prototype, the same three pre-generated stories were presented to

Table 1: *fAIble* Assessment Survey: Questions and Responses

	yes	somewhat	no
Q1) *Do story events appear to follow a coherent progression?*	44%	51%	5%
Q2) *Do the characters appear to act based on some internal reasoning and motivations?*	41%	44%	15%
Q3) *Do story events appear varied?*	32%	34%	34%
Q4) *Does the language resemble human generated narrative?*	0%	29%	71%
Q5) *Does the use of adverbs and adjectives add to the depth of descriptiveness of the story?*	20%	49%	32%
Q6) *Is the language varied across sentences and stories?*	5%	37%	59%

41 anonymous readers through an online survey hosted by a third party (sample story in fig. 3 is one of these three). The assessment focused on three aspects of how *fAIble* is perceived by readers: storyline coherence, character believability, and language quality. Respondents were asked to choose “yes”, “somewhat”, or “no” in response to six questions, which are listed in table 1, alongside the tallied results.

Although *fAIble* is a work in progress, the preliminary assessment is needed to ensure development is addressing the system’s most pressing needs first. The first three questions address story elements. Sensible story progression and goal-oriented character actions appear to be successfully implemented. The interweaved side quests are currently too similar, resulting in only “somewhat” varied story events, on average. This is to be expected given the small size of the context graph and will be resolved as we expand the number of existing action and context graph nodes. The last three questions address story language. The biggest success here was the perceived depth of the stories’ descriptive elements through the incorporation of adverbs based on character attributes and descriptions of story locations. The NLG areas in need the most improvement are language variability and human-like sentence quality. Overall, event and linguistic variation appear to need the most further development.

6 Conclusions

While *fAIble* is still in active development, much of the infrastructure is already in place. *fAIble* focuses on a modular approach, intended to capture and interconnect elements of common sense to produce believable character actions and cohesive storylines. Storyline planning is eliminated by ensuring no context is a dead end for any possible story goal. *fAIble*’s NLG focuses on enhancing story presentation with meaningful insights into character traits, reducing event redundancies, and tracking what is known to the reader.

Based on the conducted evaluation of the prototype, the immediate next steps in *fAIble*’s development are: improvements to linguistic variation (especially in thought processing), incorporating user preferences and requests for changes, expanding the context graph and the available character actions and world events, and giving agency to the antagonist as well as to the secondary characters.

Acknowledgments This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 1458272. The presented research is a contribution to the NSF Program International Research Experiences for Students (IRES).

References

Barber, H., and Kudenko, D. 2009. Generation of adaptive dilemma-based interactive narratives. *IEEE Trans. on Computational Intelligence and AI in Games* 1(4):309–326.

Dalianis, H., and Hovy, E. 1996. *Aggregation in natural language generation*. Springer Berlin Heidelberg. 88–105.

Gatt, A., and Reiter, E. 2009. SimpleNLG: A realisation engine for practical applications. In *Proc. 12th European Workshop on Natural Language Generation*, 90–93. Stroudsburg, PA, USA: ACL.

Gonzalez, A. J.; Stensrud, B. S.; and Barrett, G. 2008. Formalizing context-based reasoning: A modeling paradigm for representing tactical human behavior. *International Journal of Intelligent Systems* 23(7):822–847.

Hollister, J., and Gonzalez, A. 2017. *Cooperating Context Method: A Contextual Approach to Story Generation and Telling*. Cham: Springer International Publishing. 277–287.

Kybartas, B., and Bidarra, R. 2017. A survey on story generation techniques for authoring computational narratives. *IEEE Trans. on Computational Intelligence and AI in Games* 9(3):239–253.

Lebowitz, M. 1984. Creating characters in a story-telling universe. *Poetics* 13(3):171 – 194.

Liu, H., and Singh, P. 2002. Makebelieve: Using commonsense knowledge to generate stories. In *Eighteenth National Conference on Artificial Intelligence*, 957–958. Menlo Park, CA, USA: AAAI.

Liu, H., and Singh, P. 2004. ConceptNet — a practical commonsense reasoning tool-kit. *BT technology journal* 22(4):211–226.

Pickett, G.; Khosmood, F.; and Fowler, A. 2015. Automated generation of conversational non player characters.

Porteous, J.; Cavazza, M.; and Charles, F. 2010. Applying planning to interactive storytelling: Narrative control using state constraints. *ACM Trans. Intell. Syst. Technol.* 1(2):10:1–10:21.

Propp, V. 2010. *Morphology of the Folktale*. Univ. of Texas Press.

Riedl, M. O., and Young, R. M. 2004. An intent-driven planner for multi-agent story generation. In *Proc. 3rd Int. Joint Conf. Auton. Agents Multi Agent Syst.*, 186–193. Washington, DC, USA: IEEE.

Riedl, M. O., and Young, R. M. 2010. Narrative planning: Balancing plot and character. *J. Artif. Intell. Res.* 39(1):217–268.

Shklovsky, V. B. 1991. *Theory of Prose*. Dalkey Archive Press.

Turner, S. R. 1993. *Minstrel: A Computer Model of Creativity and Storytelling*. Ph.D. Dissertation, Los Angeles, CA, USA.

Wade, J.; Wong, J.; Waldor, M.; Pasqualin, L.; Jantke, K.; Knauf, R.; and Gonzalez, A. 2017. A stochastic approach to character growth in automated narrative generation. In *FLAIRS-30 Conf.*