Tracery: An Author-Focused Generative Text Tool

Kate Compton Michael Mateas
Expressive Intelligence Studio
Department of Computer Science University of California, Santa Cruz
1156 High St, MS:SOE3 Santa Cruz, CA 95064 USA
kcompton, michaelm @soe.ucsc.edu

ABSTRACT

Tracery is an author-focused generative text tool, intended to be used by both novice and expert authors, and designed to support generative text creation in these growing communities, and future ones. Tracery is designed to work easily in a variety of possible applications by a variety of new authors, so we present a range of existing projects that use Tracery as part of the art creation process.

1. INTRODUCTION

New communities of generative text practitioners are flourishing, with expressive mediums like Twitterbots and Twine joining existing practices of Interactive Fiction. Story grammars have been used by the academic [3] and developer [2] communities. We build on this existing history to create a tool that servers the needs of many kinds of authors. academic, game developer, botmaker, and casual. Features like data portability and ease of use are important to authors, so Tracery's scalable complexity allows for a low barrier of entry and a simple JSON file format supports a culture of sharing and remixing. So far the experiment seems to be succeeding, we are able to easily author our own work, and Tracery is being adopted by new authors as well.

2. ABOUT TRACERY

Tracery is a very small (16k) library for expanding a story grammar written in Tracery syntax into fully-generated text.

In addition to this core library, we have built (and are building more) authoring GUIs, grammar visualizations, and statistical analysis tools. The authoring GUIs help the authors create new Tracery grammars. The visualizers and analyzers take a Tracery grammar as input, and visualize information about how connected a grammar is, how deeply nested or reusable its symbols are, and what the potential space of its expanded outputs will be.

Another module is in-development to connect Tracery's text generation seamlessly to a modeled story world in a game. This allows dynamic information in the game (hero name, location, weather, current inventory) to automatically populate and update a Tracery grammar, which can then be used to generate appropriate flavor text and descriptions in game. This module is being developed simultaneously with an interactive game Neverbar so that it is informed by the *actual development needs* of a game.

Tracery, as a whole project, is an ecosystem of interoperable modules, all sharing the Tracery grammar objects as their 'lingua franca' data type. New components, tools, and visualizations can be added to the ecosystem by new developers as Tracery grows in use.

Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG 2015), June 22-25, 2015, Pacific Grove, CA, USA.
ISBN 978-0-9913982-4-9.

3. DEMOS

Our demo session highlights several works that use Tracery. Each work has generative text as a component (using Tracery), but each work also uses it in a different way.

3.1 Eternal Night Vale

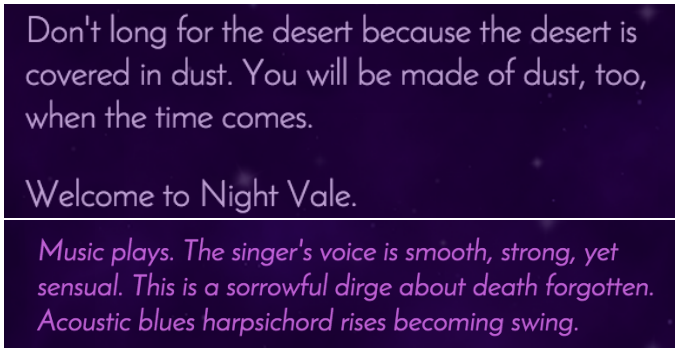

Figure 2: Generating indie music descriptions and sinister warnings in Eternal Night Vale

Eternal Night Vale was the first released stand-alone project using Tracery, and was created for ProcJam 2014, the procedural generation jam. It created possible episodes in the style of the highly-stylized podcast Welcome To Night Vale. A review in Rock Paper Shotgun praised the work as "a great little tool which condenses the tone and main elements of the show into just a few paragraph". By mimicking the structural hallmarks of Night Vale, and the unusual language of the show, and providing a graphical style that reflected the show's tone, we were able to accurately capture the spirit of the show in just a short grammar.

3.2 Interruption Junction

Figure 1: Tracery's new interactive tutorial at www.crystalcodepalace.com

Eternal Night Vale shows that Tracery can be used to create interesting valuable works, with no Javascript other than a way to write the text to the screen, but it can also be used a component of a larger game. Deirdra "Squinky" Kiai's "Interruption Junction" is the first major game released using Tracery. In their words, it is"a short one-button conversation game about being lonely in a group of people!" The player is a person in a conversation in which three other friends are discussing the activities of mutual acquaintances. The player can mash the space bar to interrupt and begin rambling about video games, but unilaterally dominating or retreating from the conversation will cause people to fade from the conversation. All of the dialogue is generated by Tracery, using a grammar by Kiai. This generative dialogue is meaningless and absurd and endless, in this case, a good fit for the theme of the game.

3.3 Beautiful Stranger

Beautiful Stranger is an interactive digital 'magic mirror'. As the user stands in front of it, a face-tracking algorithm can detect their face, procedurally overlaying their face with an emergent and artistic 'mask'. As this transformation occurs, the system begins surrounding them with a swarm of surreal over-the-top compliments. Though the focus of this project is on the generative art in the mask, Tracery allowed us to quickly create a compliment generator.

3.4 Neverbar

Neverbar is an interactive sci-fi romance story, in which the protagonist is at a time-traveling bar, and can romance a number of surreal possible love interests. The story needs to maintain consistent world state to know the gender of the protagonist, their name, their current location in the bar, their drink, how much of their drink remains, and other such story items. Neverbar is a case study, developed simultaneously with Tracery's state-maintenance module, to keep the module informed by real game development needs.

3.5 Adapting MEXICA

Since Tracery does not have its own model of story continuity, or the lagic to maintain it, it can be paired with a system that does to create generative text for a story with real narrative coherence. We test how this collaboration might work using a data structure output from MEXICA, a computational plot generator.

3.6 Cheap Bots, Done Quick

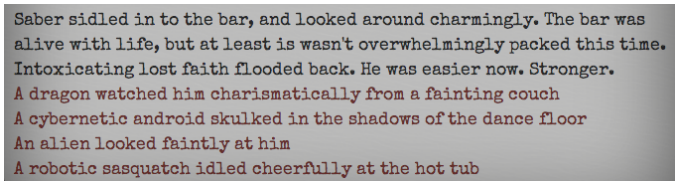

Figure 3: Neverbar generates and presents many options to the player, but must maintain world state to remember player choices.

We will also be showing selections from authors using George Buckenham's tool "Cheap Bots, Done Quick" [**?**], an online Twitterbot creation an hosting tool that was made possible by Tracery.

4. PROJECT STATUS

We began this project with the hypothesis that, if we take the *real design needs* of creators seriously, and created a usable system that addressed those needs, then creators would choose to use it in their work. "Usability" is informed by long experience in both software development and indie game development, and understanding the needs of these communities. Modularity, reliability, modifiability, portability are not just boxes to check, they are critical determiners of whether a system is adopted and used.

Several artworks and games have already been created using Tracery, by ourselves and our collaborators. Students are using it in their academic projects and independent games. George Buckenham's CheapBotsDoneQuick is introducing many novice users to bot-making. We hope to see further adoption continue in the academic community as well.

5. REFERENCES

[1] K. Compton, B. Filstrup, M. Mateas, et al. Tracery: Approachable story grammar authoring for casual users. In *Seventh Intelligent Narrative Technologies Workshop*, 2014.

[2] Orteil. Randomgen. http://orteil.dashnet.org/randomgen/.

[3] D. E. Rumelhart. On evaluating story grammars*. *Cognitive Science*, 4(3):313–316, 1980.