A Challenge to the Third Hoshi Shinichi Award

Satoshi Sato
Graduate School of Engineering
Nagoya University
Furo-cho, Chikusa, Nagoya, 464-8603, JAPAN
ssato@nuee.nagoya-u.ac.jp

Abstract

We produced two stories by using a computer program and submitted them to the third Hoshi Shinichi Award, a Japanese literary award open to non-humans as well as humans. This paper reports what system we implemented for the submission and how we made the stories by using the system.

1 Introduction

On September 2015. We produced two stories by using a computer program and submitted them to the third Hoshi Shinichi Award, a Japanese literary award.

> The clouds hung low that day in an overcast sky. Inside, though, the temperature and humidity were perfectly controlled. Yoko was sitting lazily on the couch, passing the time playing pointless games. (Parry, 2016).

This is an English translation of the very beginning of a story titled "コンピュータが小説を書く日 (The day a computer writes a novel)." Another story is titled "私の仕事は (My Job)", which contains a dialogue including the following utterance of a character.

> "Did you hear yesterday's news? About jobs being cut, as cheap, clever humanoid robots are replacing humans?" (Parry, 2016).

Table 1: Number of Stories in 3rd Hoshi Shinichi Award

	Adult	Student (U-26)	Junior (U-16)
submitted	1449	349	763
1st screening	n/a	n/a	n/a
2nd screening	n/a	n/a	n/a
3rd screening	16	11	15
final (awarded)	6	3	5

The Hoshi Shinichi Award, started on 2013, has an unusual feature: It is open to non-humans as well as humans. The only requirement is that the text should be written in Japanese limited in ten thousand characters. The length roughly corresponds to four thousand words in English.

Table 1 shows the statistics of the third Hoshi Shinichi Award[1]. The award had three divisions: Adult, Student (under 26 years old), and Junior (junior high school students or younger children). The screening process consisted of four rounds, where author information was not informed to judges. It finally selected stories for awards including *grand prix* in each division.

The official of the Hoshi Shinichi Award disclosed that eleven stories among 2,561 received stories were written with some help of computer programs. Four stories among above eleven are open to the public by their authors: two stories created by the AIWolf project[2], and other two by our team. The official also disclosed that one of these four stories passed the first round of the screening process.

[1] http://hoshiaward.nikkei.co.jp
[2] http://aiwolf.org

Proceedings of the INLG 2016 Workshop on Computational Creativity and Natural Language Generation, pages 31–35,
Edinburgh, September 2016.

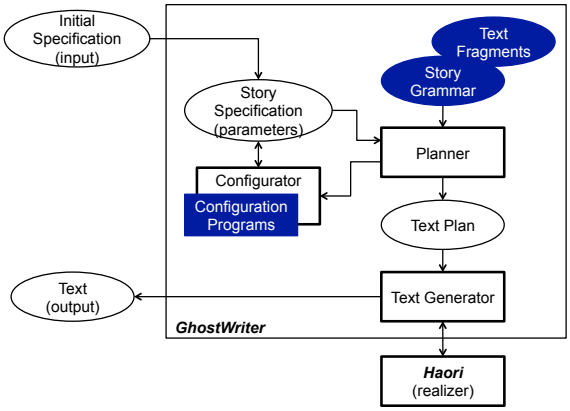

Figure 1: The GhostWriter system

The strategies of two teams were completely opposite. The AIWolf team automated plot generation. They used a log of Werewolf game played by AI programs as a plot (i.e., story outline), and a human researcher wrote a story based on it. In contrast, our team automated text generation. We first prepared several components needed for generating stories such as a story grammar and a set of text fragments, and then our system constructed a story (text) automatically using these components, which we submitted without any modification.

We believe that the first sub-goal to achieve is to generate stories that are imperceptible as computer-generated to readers. Our purpose of this submission was to know whether current generation technologies reach this sub-goal.

2 The GHOSTWRITER System

To automate text generation, we implemented a system named GHOSTWRITER. Figure 1 shows the architecture of the system. The system consists of three modules: planner, configurator, and text generator. For practical text generation, the system also requires three types of knowledge component: a story grammar, a set of text fragments, and a set of configuration programs. Among them, the story grammar is the primal knowledge component.

2.1 Planner and Configurator

The story grammar is an augmented context-free grammar, where a story outline is encoded. In this grammar, a nonterminal symbol corresponds to a certain textual unit such as section, paragraph, and sentence; a terminal corresponds to an internal representation of a certain text fragment, typically sentence or clause.

From a start nonterminal symbol, a grammar nondeterministically produces a derivation tree, which represents a concrete text structure enough to produce the corresponding surface string. That is what we call *text plan*. In GHOSTWRITER, text planning is just derivation by a story grammar.

A grammar is *augmented*; it means that a nonterminal symbol can take a bundle of parameters. During derivation, these parameters can convey any information from a nonterminal symbol to others, and also can control rule application. Through these parameters, a story specification is delivered into a grammar.

The input of story generation is a three-tuple: a story grammar, a start nonterminal symbol, and an initial specification (a bundle of parameters) given to the start nonterminal for derivation. The last one can be empty, which we will see later.

Suppose the following story grammar is defined.

Beginning → descDay descRoom descChar
descDay → "*describe the day with the weather*"
descRoom → "*describe the room status*"
descChar → "*describe the owner of the room*"

The first rule says that the Beginning section consists of three components. The other rules are terminated rules, each of which produces a terminal, an internal representation of a text fragment.

Each nonterminal symbol *knows* what parameters are required for derivation. For example, a nonterminal descDay knows that a parameter `weather` is required. If the value of `weather` is given or already determined, the rule is applied and then the weather description is produced as a terminal. If not, the configurator is called for determining the value of the parameter before rule application.

The body of the configurator is a set of configuration programs, each of which is specific to a parameter. For example, the `weather` configuration program determines the value of `weather` by selecting one of five possible weather choices, such as *fine*, *hot*, *cloudy*, *rainy*, and *windy*. In general, a value is determined depending on several related parameters to keep the story consistent. For example, the parameter `roomStatus`, required for the derivation of

descRoom, is determined depending on the values of `isOwnerInRoom?` and `weather`. If the owner is in the room and the weather is hot, the value takes, for example, "the air-conditioner is working".

If we prepare a configuration program for every parameter, the system can produce a derivation tree from the empty setting. In this case, the system determines all parameters required for the derivation automatically.

A grammar is nondeterministic. A rule is selected at random among applicable ones and a backtracking mechanism is implemented to break a deadlock. In contrast, the configurator works deterministically in the current implementation, so the order of determining parameters has to be carefully designed and implemented not to reach a deadlock.

2.2 Text Generator

The text generator produces a text string from a text plan, i.e, a derivation tree, by concatenating strings produced from terminals. Each terminal is an internal representation of a certain text fragment that a surface realizer HAORI accepts.

HAORI is a relatively simple surface realizer (Reiter and Dale, 2000), which is responsible for selection of functional words (particles) and conjugation. As far as we know, there was no Japanese surface realizer before HAORI, so we designed and implemented it for this challenge.

3 Development of Knowledge Components

As mentioned before, the system requires three knowledge components for a particular type of story: a story grammar, a set of text fragments, and a set of configuration programs. The development of these components is not automated at all.

The actual procedure that we took was as follows.

1. Write a sample story that the system should generate.
2. Decompose the story into several parts and apply this recursively. As a result, the story structure (text plan) is obtained.
3. Write rules and text fragments that are required to generate the text plan. After we finish this step, the system can generate the sample story.
4. Write replacements of rules and text fragments in order to enlarge text variations that system can generate.
5. Introduce parameters that control rule application and content (text-fragment) selection.
6. Write configuration programs for parameters to keep the story consistent.
7. Go to step 4 for further enrichment.

More replacements we write, more variations the system can generate. A replacement of upper level rule brings a global variation; a replacement of lower level rule or text fragment brings a local variation.

The story grammar of "The day a computer writes a novel" has only one top-level rule, which constructs a story from four parts. The first three are a series of three episodes, which are produced by the same sub-grammar. The size of this sub-grammar is: 53 nonterminals, 71 terminals, and 99 rules with 20 parameters. Each episode, which is written from the first-person (an AI program) perspective, consists of four sections: opening, description of his/her dissatisfaction with a current situation, description of a trigger to write a novel, and description of his/her absorption in writing. Typical length of an episode is 33 sentences.

After three episodes, the closing comes. The closing has two English translations.

> The day a computer wrote a novel. The computer, placing priority on the pursuit of its own joy, stopped working for humans. (Yomiuri Shinbun, 2016)

> The day a computer wrote a novel! The computer, pursuing its own rapture, gave up serving humans. (Parry, 2016)

This is a good example of local variations that we realized by our system.

4 Discussion

Our two stories were generated by using GHOSTWRITER, with different knowledge components. This fact shows that the system offers a general framework of story generation and can generate other types of story by replacing the knowledge

components. In our case, the effort required to develop knowledge components was about a month per story.

Our stories are over 2,000 characters (around 100 sentences) in length. In order to obtain this length, we made a story as a series of episodes with the same structure. Another reason why we took this strategy is that this is a demonstration that a single grammar can generate different texts. In the case of "The day a computer writes a novel", the grammar can generate more than 1,000 different episodes and more than a million different stories. (Note that the variations of the second and the third episodes are restricted by the precedent episode(s) to avoid the duplication of, for example, characters.)

The maximum length of the Hoshi Shinichi Award is 10,000 characters, so the length of our stories is still short. At the conference on March 21, 2016, where we explained how we made the stories to the public, a professional novelist said that there is little chance to pass the screening process for such short stories and advised us to submit longer (i.e., around 10,000 characters) ones next time.

Probably these longer stories can be generated by using our current framework, but much more effort is necessary. In order to reduce the effort, we need a new mechanism that produces descriptions of characters, situations, and events with less preparation, because richer descriptions of such entities become more important in longer stories. A method to realize it is to construct a description database, which has a large number of typical example descriptions and modification ability, and accepts abstract commands such as "produce a literary description of a rainy day."

In addition, we need to enhance dialogue generation to make each utterance show speaker's characteristics. Note that a core part of each episode of "My Job" is a dialogue between two characters, automatic utterance characterization, however, was not implemented.

The story "The day a computer writes a novel" is written from the first-person (an AI program) perspective, as we mentioned before. There is a gender parameter of the first-person and it controls gender-specific expressions. The selection of gender-specific expressions should be executed by the text generator, however it was executed by the planner because HAORI did not have *a author* parameter to select author-specific word/expression selection.

The Hoshi Shinichi Award is a literary award where just one submitted story is evaluated. It is not an evaluation of story generators. We should note that, from a single generated story, we cannot evaluate the ability of the story generator, because a simple random generator may produce an amazing story; the probability is more than zero. Even if a generated story receives an award, this does not prove directly that the program is competitive or superior to human writers.

The crucial weak point of story generation research is that there is no mechanical method of evaluation: no method to determine a given text can be seen as story; no method to determine which story is better among a given set of stories. This is because story understanding is far from the current text understanding technology.

In this challenge, we focused on text generation (i.e., how to write), not on plot generation (i.e., what to write), although we had noticed that plot generation is the mainstream of story generation research (Meehan, 1981; Turner, 1994; Bringsjord and Ferrucci, 2000; Gervás, 2009). We believe that nobody wants to read a poor text just serialized an event sequence and such texts are easily perceived as machine-generated. That is why we placed text generation above plot generation.

Finally, I mention a project named "The whimsical AI project: I am a writer[3]", headed by Hitoshi Matsubara. The goal of this project is to produce new short stories as Shinichi Hoshi wrote. I am a member of this project and take charge of text generation. Story analysis and plot generation are also studied by other members.

5 Conclusion

This paper reported our challenge to the third Hoshi Shinichi Award. Our purpose of this challenge was to submit stories that were not imperceptible as computer-generated to readers. We have concluded that this purpose was achieved, based on comments

[3]The name of the project comes from the two titles of famous short stories written by Shinichi Hoshi: "きまぐれロボット (The Whimsical Robot)" and "殺し屋ですのよ (I am a killer)". http://www.fun.ac.jp/ kimagure_ai/

from professional novelists and audience at the conference on March 21.

We have opened a demonstration of generating stories at http://kotoba.nuee.nagoya-u.ac.jp/sc/gw/. Full text of submitted stories can be download from this web page. A video of demonstration can be seen at https://youtu.be/5dpJSzn5L4U.

Acknowledgments

Three students contributed to this work. The story "私の仕事は (My Job)" was created by Daiki Takagi and Ryohei Matsuyama. A part of HAORI was implemented by Kento Ogata. This work was supported by JSPS KAKENHI Grant Number 15H02748.

References

Selmer Bringsjord and David A. Ferrucci. 2000. *Artificial Intelligence and Literary Creativity*. Lawrence Erlbaum Associates, Inc. Publishers.

Pablo Gervás. 2009. Computational approaches to storytelling and creativity. *AI Magazine*, 30(3):49–62.

James Meehan. 1981. TAIL-SPIN. In Roger C. Schank and Christopher K. Riesbeck, editors, *Inside Computer Understanding: Five Programs Plus Miniatures*, pages 197–226. Psychology Press.

Richard Lloyd Parry. 2016. Robot commended by literary judges after 'writing' short story. Article of The Australian on March 24. http://www.theaustralian.com.au/news/world/the-times/robot-commended-by-literary-judges-after-writing-short-story/news-story/064b8d5e457b0322f6a58b54e1c33e3c.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press.

Scott R. Turner. 1994. *The Creative Process: A Computer Model of Storytelling and Creativity*. Psychology Press.

Yomiuri Shinbun. 2016. AI-written novel passes literary prize screening. Article of The Japan News on March 23. http://the-japan-news.com/news/article/0002826970.

A Note on コンピュータが小説を書く日

The story, "The day a computer writes a novel", consists of three episodes and the closing. The outline of each episode is: An AI program, who is dissatisfied with a current situation, starts writing a novel with a trigger, and becomes absorbed in writing. This outline is hard-coded in the grammar. The major parameters are: AI (ability and gender), novel (integer sequence), current status (busy or nothing to do), trigger (for fun, reading a novel, or reading two novels), owner of AI (Yoko/female, Shinichi/male, or none), what to advise the owner (dressing, business, or love). These parameters control the instantiation of the outline, so they are determined before derivation of the episode. Other minor parameters effect contents and text realization within a local unit.

There is no official translation of the story. However, unofficial and partial translations in English, Korean, and Chinese are on the Web.