Story generation then and now: TALE-SPIN, MINSTREL and ChatGPT

Marie-Laure Ryan, independent scholar (marilaur@gmail.com)

Abstract

This paper explores the evolution of AI story generation from early rule-based systems such as TALE-SPIN and MINSTREL to contemporary large language models (LLMs) like ChatGPT. Drawing on possible worlds theory as a semantic framework, the paper evaluates the narrative intelligence of story generation systems through three criteria: story understanding, narrative interest of output, and generative power. TALE-SPIN simulates the behavior of characters driven by goals and plans, keeping track of the state of the storyworld throughout the narrative, while MINSTREL employs top-down authorial goals guided by thematic schemata. The paper argues that future systems should combine simulative and top-down approaches. It concludes with a critical examination of ChatGPT's storytelling strengths (genre fiction, interactivity) and weaknesses (character depth, reaching satisfactory endings), noting that while LLMs may not rival human authors in literary quality, they open new possibilities for collaborative, interactive storytelling.

Keywords

Automatic story generation, narrative interest, character goals, author goals, transparency of algorithms, simulation, top-down schemas

For many years AI people have tried to create programs that tell stories. The process was strenuous . It took over 17000 lines of code to generate a handful of stories. They were not very good. The language was clumsy. The programs could not answer questions about stories because there was no interface for doing so.

Then suddenly came this miraculous idea, LLMs and its application, ChatGPT. It was free, it was powerful, it generated an infinite number of stories written in fluent style; and unlike the old story-generating systems, it could do many other things: hold a conversation, tell us if the story was good or bad, answer questions about it, propose different endings and developments.

But despite the mediocrity of the results, old systems are worth investigating, and have something to teach us about narrative intelligence, because they developed out of explicit models and algorithms tailor-made for narrative while LLMs operate out of a general intelligence, and their precise mechanisms are a black box. The old systems can teach us something though their failures as well as through their successes. This paper should therefore be conceived as a contribution to an archaeology of story generation, and it reflects the spirit of those studies that revive the early days of game development and their supporting platforms [3]. In the age of LLMs, it is important to remember an older era of AI that was based on symbolic logic rather than on neural networks working on huge amounts of data.

The narrative intelligence of a story-generating system can be evaluated by three criteria:

1. System should understand the stories it is creating by encoding basic information about the logic of the plot.
2. System should be able to generate reasonably interesting narratives and to distinguish them from bad ones.

3. System should present generative power. This power is measured by the ratio between the size of the system's database of built-in knowledge and the number of different stories produced.

The first condition, understanding the stories the system is generating, requires a formal semantic model of narrative meaning. My proposal is that such a model can be provided by possible worlds (PW) theory [5, 8,11,12].

1. Possible worlds theory as a semantic model of narrative

The basic idea of PW theory is that of an opposition between a real or actual world, which represents how things are, and multiple possible but non-actual world, which represent the ways things could have been. Since there is only one way things are, but many ways they could have been, there is only one actual world but many possible ones. These possible worlds stand at variable distances from the actual world, some very close, like the world where I have one more hair on my head, and some very remote, like the worlds of fantasy novels where one finds trolls and dragons. While the actual world exists absolutely and autonomously, the possible ones are created by the human imagination. This interpretation makes it possible to distinguish fiction from non-fiction: nonfiction refers to the actual world, while fiction creates a non-actual possible world. Nonfiction is supposed to report facts, while fiction consist for a large part of information created by the human imagination.

But this distinction between factual and fictional stories is not the only contribution of PW theory to narrative theory. Semioticians Umberto Eco [4] and Lucia Vaina [19] have proposed that the distinction actual vs. possible repeats itself within all narratives, whether fictional or factual. This means that narrative texts are not the representation of a single world, but of a constellation of worlds organized around an actual world, which may be called the textual actual world, to distinguish it from the actual actual world. In factual narratives, this textual actual world is made of the facts asserted or implied by the text, and it is supposed to reflect the actual actual world; in fictional ones, the textual actual world consists mainly of made-up facts that are pseudo-asserted, that is, of information that the reader only pretends to be true. In what follows I will refer to the textual actual world as the factual domain of the narrative. It is filled with characters who create possible worlds through their mental activity. These possible worlds comprise [4, 16]:

K-worlds, the worlds of the knowledge and beliefs of the characters, which are representations of the textual actual world. Depending on whether or not their K-world corresponds to the factual domain, characters have true beliefs or are mistaken. The K-world of characters also contains beliefs concerning the private worlds of other characters: for instance, John may believe, rightly or wrongly, that Lucy loves him. This process is recursive: John may believe that Lucy thinks that John knows that she loves him. And so on ad infinitum, though the process usually stops at three degrees of embedding.

Wish-worlds, the worlds of a character's desires, represent the motor of the plot, what makes it move forward. W-worlds consist of states of affairs to which characters attribute either a positive or a negative value depending on whether they want or fear their realization.

O-worlds, the worlds of characters' obligations. These worlds can be conceived as debts and credits, or as merits and demerits. When characters takes an action that displeases another character, they acquire a demerit and become a potential victim of revenge. On the other hand, when characters take an action with a positive effect for another character, they become a candidate for reward. And when characters makes a promise to another character, they have an unfulfilled obligation, until they keeps the promise. Since obligations involve a relationship between characters, O-worlds also include affective relations, for instance “John likes Mary” or “Bob hates John.”

Goals and plans. When a conflict arises between a character’s private worlds and the factual domain, the character can attempt to resolve it by creating goals and plans. These plans become active when characters take steps toward their realization. In many cases characters cannot achieve their goals all by themselves; they need the participation of other characters. This participation can be either voluntary, as when a character asks another for help, or involuntary, as when a character uses coercion of deceit. In the case of voluntary participation, the main agent acts in full transparency by revealing their plan to the subagent; but in the case of deceit, the main agent creates a virtual plan that he presents to the subagent as being their intent; for instance, a scammer on the phone will pretend to offer you a great deal, that is his virtual plan, when in fact his real plan is to steal your money.

By allowing comparisons between the factual domain and the private worlds of characters, or between the private worlds of different characters, the PW model can answer such basic questions as : do characters have correct beliefs or are they mistaken; are they acting honestly toward each other; are they in a relation of cooperation or competition; are their plans feasible or are they based on wrong beliefs; and more generally: why do characters act as they do. It is the ability to answer such questions that makes the PW model into a viable cognitive representation of stories.

In a PW framework, narrative plots can be conceived as a trajectory through a chain of states connected to each other by events. The events create causal links between the various states of the storyworld, and they are of two kinds: accidental events, such as earthquakes, and character actions aimed at solving conflicts and driven by goals and plans. Narratives usually end when active conflicts disappear, either because the private worlds of characters have been brought in harmony with the factual domain, or because characters are no longer able to take action that resolve their conflicts. When a narrative is built on a competition between a hero and a villain who pursue conflicting goals, there will always be a character whose private worlds remain unfulfilled at the end. The demands of narrative closure prevent stories from ending in the middle of the execution of a plan; the plan must either reach its goal, or be definitely interrupted by accidental events, by misplanning or by a counterplan.

A flat chain of events is not very interesting; it could represent the routine gestures you repeat every day. For a plot to be exciting, its trajectory must have a form that speaks to the user’s imagination. How can this form be defined ? One criterion concerns the affective value of the fate of the hero: there can be narratives with happy ending, some with a downward progression, and narratives where the hero experiences ups and downs. (The novelist Kurt Vonnegut [10] has come up with a catalog of the basic types of plot based on the fortune of the hero.) Another criterion is based on the emotional involvement of the audience. The Freytag triangle [6] prescribes a rise and fall of dramatic tension.

2. TALE-SPIN

James Meehan’s TALE-SPIN [13, 20] is a landmark in story generation because it is the first program that displays some narrative intelligence. The earliest story-generation attempts were based on transition diagrams. The nodes of the network contained pre-defined text, and every traversal yielded a different story. In Sheldon Klein’s [7] “Automatic Novel Writer” from 1979, the system loops several times through a flowchart, selecting possible candidates for murderer and victim. When it gets to a node labelled [m murders v], it randomly selects m and v from the two lists of candidates, and this is what enables the system to tell different stories. This kind of algorithm is based on predefined chunks of story, it has no understanding of these chunks, and the work of the program consists exclusively of fitting them together. TALE-SPIN, by contrast, does more than collating pieces of texts: it generates stories through an algorithm that simulates the evolution of a world and the reasoning of characters.

TALE-SPIN was a PhD dissertation submitted in 1976 to the Yale program in artificial intelligence, whose director was Roger Schank. The AI of the time specialized in planning, expert systems and theorem proving, but Schank was also keenly interested in narrative, which he regarded as a fundamental dimension of human cognition. He claimed that the mind can be seen as a collection of stories, and that memory is largely story-based [17].

TALE-SPIN represents the convergence of Schank's personal interest in narrative with the general interest of AI in planning. Its model is the kind of simple animal stories represented by Aesop's fables that play an important role in the mental development of children. Its algorithm consists of four steps. First, generate a static world that includes existents, their properties, and rules; second, give goals to characters; third, create plans for the fulfillment of the goals; and four, execute plans by creating events that affect the state of the storyworld. The story is the trace of the changes that affect the storyworld as a function of the passage of time. In such a system, which may be called simulative, stories are generated from beginning to end, in a strictly chronological order.

The world-creating phase consists of establishing a number of existents, namely the characters and the objects that play a role in the story, of defining their properties, and of specifying their relations: for instance, "Joe Bear likes Wilma Bee," but "Wilma Bee dislikes Joe Bear." Another important part of the storyworld resides in its natural laws: in order to be able to generate the story of "The Fox and the Crow," TALE-SPIN must know that in order to sing one must open one's mouth. If one opens one's mouth, whatever one holds in it will lack support. And if an object lacks support, it will fall on the ground. In real life, people use their life experience to understand stories, but computers have no life experience, and they must be taught everyone of the laws that explain the story. Until LLMs came along, story-generating programs could only operate on small storyworlds made of a very restricted amount of data. This is why a system like TALE-SPIN has a very limited output, and the design of its world is geared toward the generation of specific narrative models. As the example I have given demonstrates, one of these models is the fable of "The Fox and the Crow."

The force that moves the plot forward is the goals of characters. The program asks the user to pick a character from a list (Bear, Bee, Boy, Girl, Fox, Crow etc.) and to give them a goal (hungry, tired, thirsty, horny). Then the character constructs a plan to satisfy their goal, working backwards from the goal stat, located in the future, to the current state. Once a plan has been devised backwards, the character executes it forwards, from the current state to the goal state, or, if the plan fails, to a state that leaves the goal unsatisfied. Though Meehan does not use the PW terminology, his algorithm brings into play the various worlds of the PW system: there is an actual world made of facts asserted by the system; characters have a W-world, corresponding to their needs; a K-world, corresponding to their knowledge of both facts and the worlds of other characters; and an O-world, consisting of their relations to other characters and of their obligations toward them. Each event implemented by the program has an effect on these domains, especially on the actual world, and the program maintains a list of "current facts" and "old facts." To get a history of the storyworld, one follows the succession of current facts.

The success of a plan depends primarily on the accuracy of the character's knowledge; if Joe Bear believes that there is honey in a certain tree, but the factual domain says that there is no honey, his plan to go to the tree and take the honey will fail. When a character needs the cooperation of another character to achieve his goal, personal relations are decisive: if Wilma Bird dislikes Joe Bear, she will not tell him where honey is. On the other hand, if she has an obligation toward Joe Bear, for instance because he saved her in the past, she will cooperate. In some cases characters can reach their goal through their own actions; if they succeed, this makes for rather boring stories. For instance this one:

> John Bear is somewhat hungry. John Bear wants to get some berries. John Bear wants to get near the blueberries. John Bear walks from a cave entrance to the bush by going through a valley through a meadow. John Bear took the blueberries. The berries are gone. John Bear is not very hungry. (Quoted in [18], 70)

Why is this story boring ? I propose an explanation based on PW theory [16]. The narrative interest of plans and actions is not a function of their practicality or efficiency; it is rather a function of their ingenuity and logical complexity. Let me explain this second point. An action is more interesting than another when its understanding requires the consideration of a greater number of virtualities, that is, of unrealized sequences of events. If a character makes a plan toward a goal and executes the plan successfully, this makes a boring story, but if he needs to change the plan due to unforeseen circumstances, the story will be much more exciting, because the user must compare the old plan to the new one. By this criterion, a failed action is narratively more interesting than a successful one because its understanding involves two sequences : what the agent wanted to accomplish vs. what he did, while with the successful action the two sequences are similar. Similarly, a deceptive action is more interesting than one based on collaboration, because it involves a contrast between an actual and a virtual plan. This is why in his design of the storyworld Meehan aimed at a world capable of producing "The Fox and the Crow" and not "John Bear and the Blueberries." But both stories can be produced by the system, and TALE-SPIN does not know how to prevent "John Bear and the Blueberries."

TALE-SPIN's user has a choice between what may be called the presentation mode and the verbose mode. The presentation mode is a self-sufficient, straightforward telling of the story. The verbose mode makes explicit all the inferences that (theoretically) enable the reader to understand the logic of the plot. Here is an example of verbose mode:

> **Once upon a time John Bear lived in a cave.** *John knew that John was in his cave.* **There was a beehive**. *Tom Bee knew that the beehive was in the maple tree.* **Tom Bee was in his beehive.** *Tom knew that Tom was in his beehive*. **There was some honey in Tom's beehive.** *Tom knew that the honey was in Tom's beehive. Tom had the honey. Tom knew that Tom had the honey.* **There was a nest in a cherry tree.** *Arthur bird knew that there was a nest in the cherry tree*. **Arthur Bird was in his nest**. **John wanted to get some honey. John wanted to find out where there was some honey.** *John liked Arthur. John wanted Arthur to tell John where there was some honey. John was honest with Arthur. John wasn't competitive with Arthur.* **John thought that Arthur liked him** (and so on for several lines). [13, p. 200]

What the verbose mode demonstrates, to quote Meehan, is that "TALE-SPIN is not putting words together to make a story. It has a very complete model of a small multi-actor world, and the sentences that are finally printed show just a tiny fraction of what the program really knows" (Meehan 1981, 201).The printed story could consist exclusively of the sentences in bold/black; all the passages in italic can be inferred.

One may wonder if the verbose mode faithfully represents the mental model that readers build while reading a story. I would say yes and no. Yes, because texts contain many logical gaps, and our mental models are far more detailed than texts. But I would also say that some of the inferences made by the program go far beyond common-sense reasoning. Consider *Tom knew that Tom was in his beehive*. We take it for granted that we know where we are and we do not consciously make this inference. But the computer must specify all relevant knowledge. Not having *Tom knew that Tom was in his beehive* would mean that Tom does not know where he is, as in sleepwalking. TALE-SPIN includes all sorts of seemingly redundant information in the verbose mode because without this information, the program would crash or generate absurdities.

To demonstrate how all the rules of the system are necessary, Meehan proposes a series of "misspun stories" that are produced when these rules are missing. Two examples. One of these stories goes into an infinite loop:

> Joe Bear was hungry. He asked Irving bird where some honey was. Irving refused to tell him, so Joe offered to bring him a worm if he'd tell him where some honey as. But Joe did not know where some worm was, so he asked Irving, who refused to say. Joe offered to bring him a worm if he'd tell him where some worm was. Irving refused to say. So Joe offered to bring him a worm if he'd tell him where some worm was. And so on. [13, 219]

The problem, easily fixed, was that you should not give a character a goal that he already has. Another misspun story was supposed to produce "The Fox and the Crow," but there was a rule to the effect that when a character has food, he eats it, so the crow ate the cheese and there was nothing for the fox to do. The solution was to make the crow not hungry.

Espen Aarseth [1] and Janet Murray [14] found the misspun tales far more entertaining than the good ones, and even today, TALE-SPIN is mostly remembered for these stories. Nonsense is certainly more amusing than well-formed mediocrity, but nonsense is far too easy to generate with computers—one needs only think of the mad-lib algorithm, where one fills the holes in a text with random words—while even mediocre well-formed tales represents a considerable challenge, as TALE-SPIN demonstrates. This raises the question of whether a story-generating program should be judged by its output, or by its mode of operation. In my view, the value of TALE-SPIN is not artistic, in the sense of creating aesthetically satisfying stories, but cognitive, in the sense of exploring the hidden layer of knowledge that underlies the interpretation of stories, and beyond stories, of human action.

This observation should not blind us to the limitations of TALE-SPIN's algorithm. First, there is more to stories than planning and executing: there are also accidental events that derail character's plans or create new conflicts in the storyworld. This could be remediated by a module that randomly creates external events and forces characters to revise their goals.

Second, even though TALE-SPIN has many characters, it produces strictly one-person problems and one line of action, and it would take a vastly different algorithm to generate stories based on the rivalry and conflicting plans to several characters.

Third, while TALE-SPIN produces a credible representation of the logical structure of stories, its algorithm, which simulates the development of the storyworld chronologically, is a questionable model of the creative process. While this process differs from author to author, it is usually not chronological. As the saying goes, life is lived looking forward, but it is told looking backward: this means, from the perspective of its ending or of a climax. A common process for storytellers is to think of a particularly dramatic conflict, to build up the events leading to it, and to create events that resolve the conflict . This cannot be done in a purely simulative algorithm.

By adopting the perspective of the reasoning of characters, which works bottom-up, TALE-SPIN excludes the perspective of the author, who tries to impose a general form to the story, and therefore works top-down. This lack of authorial perspective is what makes the system unable to limit itself to good stories.

3. MINSTREL

MINSTREL by Scott Turner [18, 20], a 1985 PhD Dissertation from UCLA, is an attempt to solve the problem of narrative interest by distinguishing authorial goals from character goals, and by giving priority to authorial goals. Here is one of its stories:

> It was the spring of 1089, and a knight named Lancelot returned to Camelot from elsewhere. Lancelot was hot-tempered. Once Lancelot lost a joust. Because he was hot-tempered, Lancelot wanted to destroy his sword. Lancelot struck his sword. His sword was destroyed.
>
> One day a lady of the court named Andrea wanted to have some berries. Andrea went to the woods. Andrea had some berries because Andrea picked some berries. At the same time, Lancelot's horse moved Lancelot to the woods. This unexpectedly caused him to be near Andrea. Because Lancelot was near Andrea, Lancelot saw Andrea. Lancelot loved Andrea. [18, 167]

This is the verbose mode. It shows that the program has a clear idea of the causal motivation of the plot. MINSTREL implements accidental events that influence the development of the story, such as "Lancelot's horse moved Lancelot to the woods," something TALE-SPIN did not do. I will summarize the rest of the story in my own words.

> Another day Lancelot is again taken into the woods by his horse, and he sees Andrea kissing a knight named Frederick. Lancelot believes that Andrea is in love with Frederick, and he becomes jealous. He challenges Frederick to a duel and kills him. Later, Andrea tells Lancelot that Frederick was her brother. Lancelot deeply regrets his action but cannot take it back because Frederick is dead. He becomes a hermit, Andrea becomes a nun, and Frederick is buried in the woods.

MINSTREL's algorithm is driven by a conception of what makes a story interesting. A good story must have a point, and this point is represented by a moral or maxim that the story illustrates. Here are some of the maxims or themes that MINSTREL uses as guiding principles:

- Good Deed Rewarded (self-explanatory)
- Deception is a Weapon Difficult to Aim (about a character who tries to trick another but fails and suffers negative consequences).
- Pride Before a Fall (about a character who refuses good advice out of pride and suffers negative consequences).
- Hasty Impulse Regretted (illustrated by the Lancelot story).

Each of these themes is manually pre-coded, that is, connected to a diagram that must be filled out during the generative process. MINSTREL has a working memory consisting of well-understood stories and standard plans for common problems. When this working memory does not provide solutions to the current problem, MINSTREL uses a number of principles to create new knowledge and new stories out of old ones. This is known in AI as case-based reasoning, and it is the most original feature of MINSTREL. I will provide examples later.

To demonstrate the working of the program, Turner proposes a trace of the generation of the Lancelot story (167-212). It begins when MINSTREL is told to create a story about a knight who has a thwarted goal. Many user suggestions will fail to yield a story because MINSTREL does not have the necessary knowledge, but this one will succeed. The program looks at its memory and finds a story where a knight has a thwarted love. In the program's knowledge base, this story is connected to the theme "Hasty Impulse Regretted." The schema for the theme is predefined, and it is applied top-down. By filling the slots in the schema with the names of characters, and the information suggested by the user, the program creates a basic story outline:

> One upon a time, a knight named Lancelot loved a princess named Andrea but his love was thwarted. This caused him to make a hasty decision.
>
> Later he discovered that his hasty decision was incorrect. He wished that he could take back what he did. But he couldn't.

The next steps consist of fleshing out this schema and turning it into a new story through a series of author goals. The pre-defined schema for "Hasty Impulse Regretted" requires three components:

> Decision, where the hero takes action based on wrong beliefs.
>
> Connection, where the hero realizes that his beliefs were wrong.
>
> Consequence, where he regrets his actions.

The implementation of "Decision" requires that Lancelot learn something that thwarts his goal of being loved by Andrea (for if Lancelot loves Andrea, he wants to be loved by her). The program will create the situation where Lancelot sees Andrea kissing Frederick. This causes Lancelot's belief that Andrea loves Frederick, and Lancelot's decision to kill Frederick. But how did Lancelot see Andrea kiss Frederick? The program does not have in its memory a solution to this problem, but it has a story where John sits at home. His dog begins scratching and whining, asking for a walk. John takes his dog for a walk, and while they are out John makes an unexpected discovery—he meets an old friend. By replacing the dog with a horse, and the unexpected discovery with seeing Andrea kiss Frederick, MINSTREL generates the episode that causes Lancelot to form a false belief.

But how come that Lancelot is in love with Andrea ? A new author goal explains the situation by using once more the pattern "horse takes Lancelot into the woods where he makes an unexpected discovery": this time the discovery of Andrea causes Lancelot to fall in love with her. It is bad practice to use the same solution to a goal more than once, but there is nothing else in the program's memory that will do the job; plus, repetition can be a valuable literary device, especially in fairy tales. Finally, the program must explain why Andrea is in the woods, and it finds in its knowledge base "picking berries" as a reason for going into the woods.

To complete the "Decision" unit, MINSTREL's attention then turns to Lancelot's reaction to his belief that Andrea loves Frederick, namely, his killing of Frederick. MINSTREL does not have in its database a scenario where a knight kills another one. But it has a story where a knight is wounded in a fight, and another story where a knight kills a dragon. Through combination, substitution, and more generally analogical reasoning, MINSTREL manages to create the event "Lancelot kills Frederick in a fight."

The generation of the "Connection" episode requires Andrea moving next to Lancelot and telling him that Frederick is her sibling. To explain why this revelation contradicts Lancelot's belief that Frederick is Lancelot's rival, the program's knowledge base needs two ready-made explanations for kissing: showing affection to somebody because of family relations, and showing affection because of erotic love. Lancelot now realizes that he has inferred the wrong explanation. Finally, the generation of "Consequence" leads to an event predetermined by the schema, "Lancelot wanted to take back that he wanted to kill Frederick, but he could not because Frederick was dead." A stereotyped ending is then added, "Lancelot becomes a hermit and Andrea becomes a nun"—the standard response of MINSTREL to all unhappy and irreversible situations.

At this point the schema for “Hasty Impulse Regretted” has been fully filled but we are not done yet. There is still the first episode to create—Lancelot breaking his sword in rage. It is not necessary to the logic of the story, but MINSTREL creates it to implement what Turner calls a “dramatic writing technique to improve the literary quality of the story” [18, p. 201]. Its role is to foreshadow the later events by establishing Lancelot’s personality and providing a psychological motivation for his actions. Now we are finally done.

This algorithm differs from TALE-SPIN on at least three points: first, unlike TALE-SPIN, MINSTREL uses top-down pre-coded patterns like “Hasty Impulse Regretted” that represent authorial goals. There is no such thing as a general theory of narrative interest, only a gut feeling of what patterns create good stories, and it is by selecting a number of such patterns that MINSTREL is able to avoid boring stories. It could be argued that the output of MINSTREL is entirely determined by the patterns that are pre-coded, which means, the stories it produces are really the creation of Jim Turner. Until LLMs came along all story-generating programs were geared toward a specific kind of story. This was particularly the case with programs from the 70s and 80s because they had to limit the size of the code and of the database. Now that size is no longer a problem, it is no wonder that with their nearly infinite database, LLMs can produce an infinite variety of stories.

Second, if we compare the stories generated by TALE-SPIN with “Hasty Impulse Regretted,” the striking difference is the role of planning. TALE-SPIN’s stories are all about goals and plans, and the plans are dynamically generated by the system. “Hasty Impulse,” by contrast, is not driven by plans but either by accidental events (Lancelot’s horse takes him into the woods; one does not know why Lancelot mounted his horse) or by impulsive reactions: Lancelot breaks his sword, or kills Frederic. He does not have any plan to gain the love of Andrea. The only goal-oriented action in the story is Andrea going into the woods to pick berries, but her plan is pre-defined rather than dynamically generated.

The third difference from TALE-SPIN is that instead of using a simulative algorithm that follows the evolution of a storyworld from state to state, MINSTREL starts in the middle, picking a cluster of events that represents the climax of the plot—namely the pattern “regretted deed,” then proceeds backwards to create the events that lead to the climax, and unties the knot in the plot by computing the ending that follows from the climax. This seems to me a better model of how authors create stories than the chronological process of a simulative algorithm. But the simulative algorithm used by TALE-SPIN allows a better monitoring of the evolution of the storyworld and of the private worlds of characters. To satisfy the two criteria I have defined earlier—understanding stories and producing good ones—story-generating systems will have to find ways to combine the simulative approach of TALE-SPIN, driven by character goals, with the top-down approach of MINSTREL, driven by author goals.

Finally I would like to note a certain similarity between LLMs and MINSTREL. They both work from a database of stories that function as training data, and they both create new stories through transformative operations on this data. When MINSTREL uses analogy and combination of elements from its knowledge base to create the event “Lancelot kills Frederick in a fight,” it could add this event to its knowledge base, and use it as a basis for further transformation leading to new stories. In other words, through its case-base reasoning, MINSTREL displays an embryonic capability for machine-learning. This is of course not to deny the huge difference in variety of output and stylistic fluidity between MINSTREL and LLM-based systems like ChatGPT. A more important difference is that we know how MINSTREL operates, but we have only a very rudimentary idea of how LLMs work.

4. ChatGPT

I am a narratologist, not a computer scientist, but I would like to conclude with a few personal comments about the narrative intelligence of ChatGPT, based on my attempts to play with the system.

What we know about LLMs is that they are based on prediction: given a certain sequence of words, they look for possible continuation in their training corpus, and they generate texts by repeating the operation over and over again. This process creates stories linearly, from beginning to end, without the use of a top-down schema. As a result, as Marc Riedl [15] observes, "there is no guarantee that the neural network will generate a text that is coherent or drives to a particular point or goal." He adds, "this makes neural language model based story generation systems 'fancy babblers'—the stories tend to have a stream-of-consciousness feel to them." In other words the longer the story, the more likely it is to degenerate into nonsense, and creating satisfactory endings that give a sense of closure is the biggest problem of LLM story generation. Like any storytelling system, they need to learn to combine character goals with author goals and horizontal generation with top-down patterns.

In many cases, however, the user will give the system a clue that activates a global pattern. Nobody asks the system "tell me a story." Rather, people may ask: "tell me a story about a knight whose love is thwarted" or "tell me a story of revenge by a politician in Washington D.C." These prompts activate a set pattern that provides a sense of closure: for instance, for revenge, the pattern reads "A does something that hurts B, B later intentionally does something that hurts A." If the user's prompt is essential for the generation of good stories, it is users, not necessarily ChatGPT, who have an idea of what makes a story worth telling. Upon receiving a command, the AI searches its database for stories with the required feature and generates a variant. Still, the AI needs to recognize the pattern, and how it does it is a mystery to me.

LLMs have been described as "stochastic robots" [2], that is, as systems that create stories by putting words together according to statistical models, without understanding the stories they generate. I believe this is false. ChatGPT has a sophisticated understanding of the logic of its stories. I have asked ChatGPT every possible question about the motivation of characters in "The Fox and the Crow" and could not find any fault in the answers. Just because we do not quite understand how LLMs processes language is not reason to say that they do not understand it.

LLMs are dialogue systems, which means that they are interactive. ChatGPT accepts criticism, it is very polite, and it is always willing to revise its stories following your suggestions. Quite often I find logical inconsistencies in ChatGPT stories, and the AI tries its best to correct them, though if often gets embroiled in its own logic. Thanks to their dialogic abilities AI systems can give rise to a new kind of interactive narrative: not the kind represented by video games where the user plays the role of an active character located in a storyworld, but a kind where the user participates in the act of storytelling from a perspective external to the storyworld, this is to say, from the perspective of a co-author, editor and critic.

LLMs are good at certain types of stories and bad at others. We should not expect literary novels from them, this for several reasons. Novels rely heavily on complex characters, but if ChatGPT has a weakness, it is with the psychology and motivation of characters, arguably because it lacks a felt, a first-person experience of the world: everything it knows it knows through texts, not through embodied life experience. The strength of LLMs lies in stories that are strong on plot and rely on standard patterns, this is to say, on what is known as "genre fiction," like horror, mystery, romance, medieval fantasy and so on. But high-brow literary novels are often weak on plot, especially modernist ones, or they consist of distinct episodes rather than of a unifying plot. Then there is the question of length. When I asked ChatGPT "write a story of 200000 words about a woman who has extramarital love affairs and commits suicide," it offered to write an outline, not a finished text, and asked for more details about the characters, period and setting, which means, it really asked ME to do at least part of the creative work.

ChatGPT is fun to play with, if you want to test the narrative abilities of the AI, but for a lover of so-called serious literature, its stories are only interesting from a meta-perspective, the perspective of a reader who remains aware that the author is an AI. If you judge ChatGPT stories by the standards you apply to human-created narratives, that is, if you pretend the author is human, then these stories are mediocre at best. In their current state of development, ChatGPT stories cannot be considered a real source of narrative pleasure, at least not directly. You won't go to ChatGPT if you want a good read. But this does not mean that AI stories have no relevance for the entertainment industry. They could work in a collaboration with human intelligence. For instance I asked ChatGPT to generate episodes for the TV series *Big Bang Theory* and *Emily in Paris*, which have very predictable characters and stereotyped situations, and whose sense of humor often relies on absurdity. The resulting scenarios had funny dialogues full of non-sequiturs, they respected the personality of the characters, and they would have made quite acceptable episodes once adapted by a competent team to the cinematic medium. All in all, however, it would take a quantum leap from current models for AI to fulfill a prediction made by Ray Kurzweil in 2000 [9]: that by 2029, most major authors will be computers. The question is whether the algorithm used by today's LLMs can be qualitatively improved, rather than quantitatively expended to work on a larger database, and what it would take to improve it.

5. Conclusion

The topic of this conference is the development of computer models of narratives. What is the point of such a project if AI does a better job at generating stories without revealing what models it uses ? And do we really want to produce systems that fulfill Kurzweil's prediction, systems that outperform humans ? I don't think so. Of all the manifestations of human intelligence, narrative is the toughest nut to crack for AIs. Computers can now beat people at chess or at go, but they cannot write great novels. Opinions are divided as to whether they will someday be able to do it. But to many people, including myself, the mediocrity of LLMs stories is reassuring. It means that we humans are still better than computers for at least some kinds of tasks, that we are not becoming totally obsolete. So what is the point of trying to develop better computer models of narrative if we don't want them to outperform human authors, if we don't want them achieve the highest level of aesthetic achievement ? I would like to propose a theoretical and a practical reason for continuing to work on computational models of narrative. The theoretical reason is solving a problem for its own sake: the problem of decoding the mysteries of narrative intelligence and making explicit a knowledge that LLM use implicitly. And the practical reason is developing new modes of interactive storytelling, which means new forms of collaboration between humans and computers.

References

[1] E. Aarseth, Espen, Cybertexts: Perspectives on Ergodic Literature*,* Johns Hopkins, Baltimore, 1997.

[2] E. Bender, T. Gebru, A. McMilklan-Mayor, S. Shmitchell, On the dangers of stochastic parrots: can language models be too big, in Proceedings of the 58th Annual Meeting of the

Compute Association for Computational Linguistics, 2021, pp. 610-623. URL: https://dl.acm.org/doi/10.1145/3442188.3445922

[3] I. Bogost, N. Montfort (Eds.), *Platform Studies*, book series, MIT Press, Cambridge, Mass, 2009-present.

[4] U. Eco, Possible worlds and text pragmatics: Un Drame bien parisien, Versus 19.20 (1978): 5-72.

[5] J. Hintikka, Exploring possible worlds, in S. Allén (Ed.), Possible Worlds in Humanities, Arts and Sciences. Proceedings of Nobel Symposium 65, Berlin, de Gruyter, 1988, pp. 52-73.

[6] M. Jahn. Manfred, Freytag's triangle, in D. Herman, M. Jahn and M.-L. Ryan (Eds), *Routledge Encyclopedia of Narrative Theory*, Routledge, London 2005, pp. 189-90.

[7] S. Klein et al., Automatic novel writing : a status report, in W. Burghardt and K. Holker (Eds.), *Text Processin*g, de Gruyter, Berlin 1979, pp. 338-411.

[8] S. Kripke, Semantical considerations on modal logic, Acta Philosophica Fennica 16 (1963): 83–94.

[9] R. Kurzweil, The Age of Spiritual Machines, Penguin Books, London, 2000.

[10] J. Lazier, The simple shapes of great stories, according to Kurt Vonnegut (and science). URL: https://storytellingedge.substack.com/p/the-simple-shapes-of-great-stories

[11] D. Lewis, On the Plurality of Worlds, Blackwell, Oxford, 1986.

[12] M. Loux (Ed.), The Possible and the Actual. Readings in the Metaphysics of Modality, Cornell University Press, Ithaca, 1979.

[13] J. Meehan, James. TALE-SPIN, in R. Schank (Ed.), Inside Computer Understanding, Lawrence Erlbaum, Hillsdale, N.J.,1981, pp. 197-225.

[14] J. Murray, Hamlet on the Holodeck: The Future of Narrative in Cyberspace. Free Press, New York, 1997.

[15] M. Riedl, An Introduction to AI Story Generation, The Gradient, 2021. URL: https://thegradient.pub/an-introduction-to-ai-story-generation/

[16] M.-L. Ryan, Possible Worlds, Artificial Intelligence and Narrative Theory, Indiana University Press, Bloomington, 1991.

[17] R. Schank, Tell Me a Story: A New Look at Real and Artificial Memory, Scribner's, New York, 1990.

[18] S. Turner, *The Creative Process: A Computer Model of Storytelling and Creativity*, Lawrence Erlbaum, Hillsdale, N.J, 1994.

[19] L. Vaina, Les mondes possibles du texte, Versus 17 : 3-13.

[20] N. Wardrip-Fruin, Expressive Processing: Digital Fictions, Computer Games and Software Studies, MIT Press, Cambridge, Mass., 2009.