Stories of the Town: Balancing Character Autonomy and Coherent Narrative in Procedurally Generated Worlds

Chris Miller
Department of Computer Science
North Carolina State University

Mayank Dighe
Department of Computer Science
North Carolina State University

Chris Martens
Department of Computer Science
North Carolina State University

Arnav Jhala
Department of Computer Science
North Carolina State University

ABSTRACT

Procedural narrative generation systems often focus on autonomous agent based simulations to create emergent interactions, plan-based approaches to provide guarantees for coherence, or using elements of simulation to guide plan-based approaches. These different approaches, with some exceptions, tend to trade off character autonomy in service of more designer controlled experiences or content authoring in service of encoding domain knowledge of possible branches of the narrative and participating characters. We have developed a system, called Stories of the Town, that automatically generates narratives by synthesizing three distinct approaches to traditional narrative generation: context-free grammars, planning, and simulation. More specifically, our system generates narratives via probabilistic context-free grammars applied to state-space planning problem solutions from planning problem formulations of simulated character models. Our system uses character simulations to generate variety in narratives and ensures narrative coherence through authoring probabilistic context-free grammars. By doing so, this system takes advantage of the strengths of each individual approach (e.g. controllability, scalability, intentionality, and variety) to generate narratives that are extensible, expressive, consistent with simulated character personalities and histories, and controllable. We show that this system has strong potential in automatically generating varied, complex, consistent, and goal-oriented narratives. Further development of the system will allow for more efficient utilization of the strengths of each narrative generation approach while also using these strengths to supplement their individual shortcomings.

CCS CONCEPTS

• **Applied computing** → **Arts and humanities**; • **Computing methodologies** → *Modeling and simulation*; • **Software and its engineering**;

ACM Reference Format:
Chris Miller, Mayank Dighe, Chris Martens, and Arnav Jhala. 2019. Stories of the Town: Balancing Character Autonomy and Coherent Narrative in Procedurally Generated Worlds. In *The Fourteenth International Conference on the Foundations of Digital Games (FDG '19), August 26–30, 2019, San Luis Obispo, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3337722.3341850

FDG'19, August 26-30, 2019, San Luis Obispo, CA, USA

ACM ISBN 978-1-4503-7217-6/19/08…$15.00
https://doi.org/10.1145/3337722.3341850

1 INTRODUCTION

In this paper, we set out to procedurally generate varied stories involving characters that exercise their autonomy within a procedurally generated world while still maintaining story coherency. Our system, Stories of the Town, aims to do this via planning-based narrative generation with agents extracted from a social simulation involving simulated characters with complex relationships and preferences. In short, Stories of the Town generates narratives via a probabilistic context-free grammar (PCFG) applied to the solution steps of planning problems crafted from complex simulated characters.

Stories of the Town, combines three distinct approaches to narrative generation, namely context-free grammars, planning, and simulation, to generate narratives in a way that is extensible, expressive, consistent, and controllable. Stories of the Town capitalizes on the advantages of each individual method in a cohesive manner that also circumvents some of the shortcomings of each respective method. The simulation component of the system is built upon Talk of the Town, a robust and extremely detailed open-source [1] simulation of fictional characters existing within a simulated town and their interactions with each other [9]. The planning component of the system reimplements Glaive, a state-space narrative planner [12]. Lastly, the context-free grammar (CFG) portion of our system is inspired by Tracery, a narrative generation/authoring tool that allows users to generate narratives by authoring grammars describing potential stories [3].

Stories of the Town begins its narrative generation process by first running an instance of the Talk of the Town simulation. Once the simulation has produced a sufficient amount of data (i.e. character models, relationships, etc.), this data is extracted and reformulated as a planning problem description compatible with Glaive. Glaive then returns a problem solution to the aforementioned planning problem which is subsequently reformulated as a coherent, semantically well-structured story via our PCFG. The simulation-based component of the system allows us to add variety and a degree of randomness to the actors in our generated narratives. Meanwhile, Glaive allows us to produce coherent solutions to planning problems based on extracted, simulated character models despite the randomness derived from the simulation component. Generated problem solutions are then

[1] https://github.com/james-owen-ryan/talktown

transformed into a controlled semantic structure through our PCFG. Another unique benefit of using Glaive as opposed to a standard plan solver is that of agent intentionality in planning problems.

2 RELATED WORK

2.1 Talk of the Town

The simulation component of our system is essentially what creates the content for our generated narratives. This component is an extended version of Ryan et al's Talk of the Town. Talk of the Town simulates a small American town occupied by simulated characters who form and propagate subjective knowledge of the simulated world, engage with social interactions with other characters, pursue careers, build families, have detailed personalities and relationships, and much more. The creators of Talk of the Town also created an authoring tool for in-game text generation called *Expressionist* that has been used alongside Talk of the Town in the context of dialogue generation [8]. However, this case study of Talk of the Town was not used to generate complete narratives. It is here that our work diverges.

We have chosen Talk of the Town for our simulation component due to the richness and variety of the characters and subsequent relationships it is able to generate. Talk of the Town generates complex character information and relationships that we believe can be used to generate interesting stories. This information includes but is not limited to familial relationships, friend/enemy relationships, occupations (incl. information such as place of work, boss, workplace acquaintances, etc.), love interests, and personalities. Previous work has shown that conflict is a key component in the evaluative process of potentially interesting stories and we believe that the aforementioned information from Talk of the Town can be used to make interesting stories from a conflict perspective [11]. However, doing so will require a coherent story resulting from the actions and motivations of characters. It is here that Glaive—an intention-based state-space narrative planner—is particularly useful [12]. As we will show later, we use Talk of the Town to produce characters for generating the initial and goal states for Glaive's planner. Additionally, we keep track of character information (e.g. personality, relationships) from Talk of the Town which we reference for selecting probability distributions for the PCFG. This is the current extent to which we are using Talk of the Town.

2.2 Glaive

A common way to generate narratives is via planning [10]. Planning for narrative generation is rooted in the idea that formulating a story is akin to a planning problem in which one wants to search through a plan space to ultimately reach an authorial goal state from some initial state. Planning-based narrative generation has also been previously used for narrative generation within simulated game universes to add a reasoning element to non-playable character (NPC) actions to help them decide how to take actions and infer about other NPC's while adhering to specific plans [2].

Glaive is a state-space planner for narrative planning problems designed to solve the intentional planning problem as described by Riedl et al [6, 12]. Glaive reasons about character intentionality while executing state-space heuristic search planning algorithms to solve narrative planning problems as described by an initial problem formulation. Intended narrative goals (i.e. intentions) of agents are described in an initial problem formulation which also contains information about the initial state of the world and agents. In the context of Glaive, intentions represent desired narrative outcomes of *agents* in a planning problem but, unlike goals, do not have to be ultimately realized in a problem solution. This distinction is important in that it allows for intention-oriented conflict between agents (e.g. multiple agents may intend to possess a particular item) while still being able to satisfy narrative goals.

Glaive generates action sequences to solve narrative planning problems via two distinct inputs: an initial problem definition and a world domain definition. The problem definition contains problem instance specific information consisting of world constants, the initial state of the planning problem represented as a set of initial predicates akin to The Planning Domain Definition Language (PDDL), and a set of states that collectively define a narrative goal for that problem instance. The domain definition specifies the types of objects that can exist in problems, additional constants, and most importantly, the actions available to agents described as a set of preconditions and effects and whether or not an action must be intended by an agent. These domain rules are manually authored and, in our case, are designed to encourage actions that we would like to see agents perform in generated narratives.

2.3 Tracery

Tracery is described as an author-focused tool for generating text and narrative with the explicit intent of being lightweight and syntactically simple [4]. Tracery is built upon a context-free grammar that generates text via expansion rules governing how to rewrite text around grammatical symbols indicated by hashtags. Following their given example, #animal# expands the *animal* symbol according to the following example production rule: *animal : ["panda", "ocelot", "meerkat"]*.

In Tracery, #animal# would be rewritten as either "panda", "ocelot", or "meerkat". Tracery also supports recursion to arbitrary depths so production rules like the following are also valid: *pet : ["puppy", "#animal#", "kitty"]*.

Two of the main benefits of using Tracery and subsequently context-free grammars for generating narratives are the ability to be able to generate narratives of large complexity via simple, yet powerful rules and strict control over the structure and flow of generated narratives [1, 7]. However, one notable drawback of this approach is that the system designer does not have any control over picking a particular term for an expansion rule to follow. In Tracery, expansion rule terms are selected with equal probability. Furthermore, using context-free grammars for text generation comes with a significant authorial burden in that it is the author's responsibility to account for modularity of their production rules, the interestingness and variedness of different story branches, and maintaining a coherent consistency of actions particular to different agents. Additionally, as an author's story becomes longer and more complex, revising parts of the story to maintain consistency can take significant effort and time. It is for these reasons that we have developed a PCFG outside of Tracery supplemented by a system containing components capable of ameliorating some of the drawbacks of the context-free grammar approach to narrative generation.

3 APPROACH

This section details Stories of the Town's approach towards using Talk of the Town, Glaive, and PCFGs to generate narratives. Figure 1 provides a graphical representation of Stories of the Town's narrative generation process. As this section will also show, a significant portion of our work focused on deciding how to transform and utilize information from Talk of the Town to construct narrative planning problems for Glaive.

3.1 Generating Characters and Content

Our narrative generation process starts with running our modified version of Talk of the Town. Talk of the Town simulates years and generations of people up until a prespecified number of years and then shows key relationships produced during the simulation including love triangles, cases of unrequited love and more [2]. For our preliminary study with our modified version of Talk of the Town, we halt the simulation after 60 years and track simpler, more specific relationships that we are interested in using for our stories. An example of general character relationships that we are interested in are the *like* and *dislike* relationships between characters. However, due to the random nature of the simulation, there is no guarantee that each simulation of 60 years will actually produce the data we seek. As such, our modified version reruns the simulation until a specified number of general relationships are produced. The complexity of these prespecified relationships can also affect the length of the generated narratives given that more complex goals derived from certain relationships might require more actions to be satisfied.

As Talk of the Town runs, it maintains a list of all the living characters in the simulated world as *Person* objects containing references to other characters according to one's familial and social relationships as well as one's personality, occupational, and general biographic information. When our simulation process halts, we export these *Persons* as string lists containing specific information we might be interested in incorporating into our generated narratives. Some examples of the data that we extracted in our preliminary study are names, occupations, lists of friends, lists of enemies, spouses, and personalities (incl. extrovertedness, introvertedness, openness to experience). We have chosen to *initially* focus on these facts since they are simple enough to capture character traits and relationships for our narratives. Even though these are simple traits, Talk of the Town is complex enough to still provide a good range of varied characters and we have already constructed a framework for storing additional character information that we could access later if necessary.

3.2 Generating Problem Definitions

After producing easily accessible objects containing specific information about simulated characters, the next step of our narrative generation process is transforming our extracted data into an appropriate problem definition for Glaive to solve. Describing an initial problem definition requires defining a series of object types for object instances in a problem, creating a series of predicates that describe the initial state of the problem, specifying any possible intentions of characters, and defining the predicates that describe a desired goal state. Listing 1 shows an example initial problem definition with some predicate omissions for brevity:

```
(define (problem tot)
 (:domain tot-domain)
 (:objects Patricia Gary - character)
 (:init
   (at Patricia Airport)
   (introvert Patricia)
   (likes Patricia Gary)
   ...
   (item_at flower Park)
   (intends Richard (likes Patricia Richard))
   (intends Richard (persuadedtodislike Richard))
 (:goal
   (and (loves Gary Patricia)
   (likes Patricia Richard)
   (persuadedtodislike Richard)))
```

Listing 1: An example PDDL problem definition for characters from Talk of the Town. It initializes the locations of characters and items, character properties, and intentions. The intentions here state that Richard will try to get Patricia to like him and that Richard will try to persuade someone to dislike someone else (not himself). It also defines three narrative goals to be achieved which are revisited in Section 3.4.

In order to pick the characters for a generated narrative, our system randomly selects a prespecified number of characters that collectively contain specific amounts of information. For example, one selection routine is to repeatedly select five characters at random until there are at least 2 like, 3 dislike, and 1 worst enemy relationships among these characters. We have set up our requirements like this in order to start introducing simplified forms of conflict and preconditions for actions later described in our world domain. Once a set of characters has been selected that also satisfies our specific conditions, we then transform the characters' attributes into PDDL predicates via straight-forward text parsing. In order to prevent an abundance of unused predicates in our problem definitions we only transform attributes relevant to the possible preconditions and effects of actions later described in the world domain. This includes character personalities as well. For example, if a character's occupation is never utilized or necessary to know for a specific problem or action, then it is not transformed into a predicate. We have designed Stories of the Town to do this predicate "filtering" based on whether or not the predicate is predefined in Glaive's planning domain. If the predicate is not defined in Glaive, it is not included since Glaive will not reference the predicate for any action preconditions or effects. Performance-wise, it is also best to not include superfluous predicates. This is particularly important because Glaive, as a state-space planner, will take considerably more time to enumerate all possible states as our set of initial problem predicates grows. Hence, it is imperative, and an ongoing challenge suitable for future work, to carefully select the minimum number of predicates that can produce solvable problems while still resulting in interesting narratives.

There are also some predicates that when added to our problem definition also add an intention for that character. Namely, the existence of some particular predicates for a character also introduce motivations for particular actions/outcomes that character intends to

[2] Character names are randomly generated and any similarities with existing people are purely coincidental.

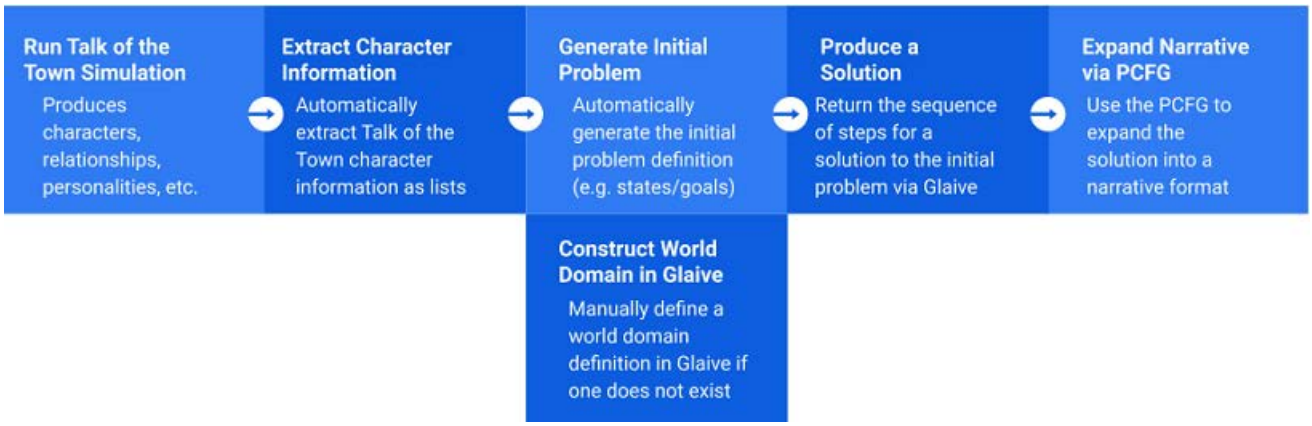

Figure 1: Execution flowchart for Stories of the Town that traces how it generates narratives via its three distinct components.

achieve. One clear example is that of worst enemies. If Person A is marked as the worst enemy of Person B and both persons are defined in the problem definition, then Person A has an intrinsic motivation (i.e. intention) to persuade one of Person B's friends to dislike Person B. This intention, combined with other intentions and goals of Person B or other characters, introduces conflict by undercutting other characters' plans and individual goals. A common goal that we utilize is for a character to get another character to like or love them and persuading that character to dislike the character pursuing this goal interferes with this. This conflict can also come in the form of multiple characters competing over items that are required for performing certain actions needed to achieve particular goals.

3.3 Defining the World Domain

The second key part of the Glaive planner is the world domain knowledge. As briefly mentioned earlier, the domain defines world constants, object types, predicates, and actions available to agents. In our system, the world domain contains characters, items, and places as object types. Characters, items, and places are predefined as object types in our domain whereas instances of these objects are typically defined within the initial problem definition. Predicate definitions simply define the structure of predicates and what object types are included in them. For example, *(likes ?character - character ?characterb - character)* defines the *likes* predicate as being between two characters: *character* and *characterb*. Since Talk of the Town does not assume that all relationships are intrinsically reflexive (as evidenced by cases of unrequited love) our predicate definitions imitate that unreflexivity.

Actions are described akin to PDDL actions in that actions require a set of preconditions to be fulfilled and produce resulting effects. In Glaive, actions also have parameters which define the input objects and their required types and an *agents* attribute which is a flag for intentionality. Namely, the *agents* attribute marks which agent is intending for the action to be executed. Listing 2 describes the *give* action we have defined. This describes an action in which a character gives another character an item thereby making the receiving character like (and subsequently no longer dislike) the character giving the item.

```
(:action give
 :parameters (?giver - character
   ?recipient - character
   ?item - item ?place - place)
 :precondition (and
   (not (= ?giver ?recipient))
         (has ?giver ?item)
         (at ?giver ?place)
         (at ?recipient ?place))
 :effect      (and (not (has ?giver ?item))
         (likes ?recipient ?giver)
         (not (dislikes ?recipient ?giver))
         (has ?recipient ?item))
 :agents      (?giver))
```

Listing 2: An action called *give* defined in our world domain. The preconditions check that the giver is not the recipient, the giver has the item, and that the giver and recipient are at the same place. This action produces the following effects: the giver no longer has the item, the recipient likes the giver, the recipient no longer dislikes the giver, and the recipient has the item.

Other actions in our current system include traveling from one place to another, picking up an item at a current location, persuading a character to dislike another character, and stealing an item from another character, and more. It was mentioned earlier that limited quantities of particular items are used as a simple way to introduce conflict among characters with competing goals and the *steal* action builds upon this by allowing a character to steal items from characters that they do not like.

One of the greatest strengths to this plan-based approach for generating narratives is that the world domain is defined independently of the initial problem definition. A world domain can be applied to several different problem definitions without needing to change the domain knowledge; assuming that it is descriptive enough to be generally applicable to the types of problems that might be generated. This is particularly important to us because this separation of the domain knowledge and problem definition greatly enhances the modularity of our system in regards to being able to generate narratives from various problem definitions derived from a randomized

simulation. Even though these domain rules must still be manually authored, one can directly focus on the types of actions that they might like to see agents perform throughout the narrative. In summation, we use Glaive to yield solutions to a large class of problem definitions which we can then transform into natural language sentences via our PCFG.

3.4 Problem Solution to PCFG Generated Narrative

The last step of our narrative generation process involves using our PCFG to turn problem solution action sequences into natural language sentences. Glaive returns solutions to problems as a sequence of actions that agents take at discrete time steps to yield the desired goal state(s). Listing 3 shows a solution to the problem defined in Listing 1.

```
(persuadetodislike Richard Patricia Gary Airport)
(retrieve Richard food Airport)
(give Richard Patricia food Airport)
(travel Patricia Airport Studio)
(retrieve Patricia brochure Studio)
(dance Patricia)
(travel Patricia Studio Airport)
(flirt Patricia Gary Airport)
```

Listing 3: Problem solution for the problem described in Listing 1.

Listing 3 achieves the three goals from Listing 1: Richard persuading someone to dislike someone else, Patricia liking Richard, and Gary loving Patricia. In the first step of Listing 3, Richard persuades Patricia to dislike Gary at the airport. This action satisfies the first goal mentioned above. Richard then retrieves food from the airport which is then given to Patricia to satisfy the second goal of Patricia liking Richard. Patricia then travels to the dance studio to retrieve and then use the dance brochure in order to become an extrovert. Our decision to make the dance action turn someone into an extrovert is arbitrary in that we could have defined any number of other actions to produce this effect as well. Regardless, given that being extroverted is a precondition to flirting, Patricia, an introvert, had to perform this action in order to flirt with Gary to satisfy the third goal of Gary loving Patricia. The solution ends here given that all of the goals from Listing 1 have been satisfied.

We then pass the solution as input to our PCFG which maps each action to a specific semantic structure output (i.e. sentence with variable parsing) with character names, items, and locations as variable terms. The probabilistic component of our PCFG refers back to the lists we created for each individual character in order to reference a character's personality. The character's personality attributes (e.g. introvertedness, openness to experience) are used as variables to change the stylistic aspects of the generated sentences; therefore allowing for CFG term replacement that is dependent on individual character personalities. We apply a weighted distribution to the probability of certain terms being selected that depends on a combination of various personality factors such as whether or not a person is extroverted, open to experience, and more. Hence, unlike Tracery, term selection is not assumed to operate on an equal probability distribution. These personality traits are derived from the Talk of The Town simulation, and we use these traits to manually author the weighted probability distribution. In the current state of Stories of the Town, although we consciously assign the relative values of each expansion term's probability, the exact values are arbitrary. In other words, for an extroverted character, although an extroverted outcome might be assigned a probability of 0.70 and an introverted outcome assigned a probability of 0.30, the exact values of 0.70 and 0.30 are arbitrary.

Having individual personalities affect the types and style of sentences used in our generative grammars adds a noticeable degree of variety while allowing our PCFG to be modular in that our PCFG can be discretized into the parsing of possible actions and their input parameters rather than having to use a CFG to maintain an entire story. However, we still retain the ability to expand upon the generated semantic structure of the actions that we parse and the arbitrary recursion depth capabilities of CFGs.

4 RESULTS

4.1 Tracery Output

```
Charlene Follin eagerly decided it was time to win
the heart of Harold Wank. As such, Charlene Follin
biked to Sunset Bistro to meet Harold Wank. Once
Charlene Follin arrived at Sunset Bistro they
picked up a burrito to give to Harold Wank. It
turned out however that Charlene Follin's evil
rival Gilbert England was there. Gilbert
England cunningly blew up Charlene Follin's gift
for Harold Wank and then fled. Charlene Follin was
sad but was determined to get Harold Wank to like
them. So Charlene Follin bought a ring from a
department store. Charlene Follin eventually found
Harold Wank at Sunset Bistro and admitted their
feelings to Harold Wank and presented a gift.
Harold Wank happily accepted Charlene Follin's
gift and hugged Charlene Follin.
```

Listing 4: Example of a short story generated with Tracery.

Listing 4 shows a simple, manually authored story generated in Tracery. This story is one of several possible stories generated by its specific CFG which consists of about 14 different grammar symbols integrated into the structure of the generated story defined as sentences containing expansion rules for symbols. Considering the various possibilities for each symbol, this story is 1 out of 16,796,160 possible stories. Essentially this resulted from the possible values that each reference to a grammar symbol may assume in the CFG.

As mentioned in the overview of the CFG approach to narrative generation, CFGs as utilized in Tracery place nearly all responsibility of generated narratives on the author. In this CFG approach, the author essentially authors all aspects of the story. The context, structure, variety, and basically all other aspects of the story are handcrafted by the author. Variety results from the author providing more and more alternative expansion rules (i.e. replacement rules) for grammar symbols. Complexity can be added to these generated stories by nesting symbols within symbol expansion rules but whenever an author decides to significantly change the story structure,

revising these rules to fit within a constantly changing story can require a significant amount of time and effort.

Causality within the CFG also requires skillful use of grammar expansion rules in that causality requires nesting grammar symbols within symbols. Nonsensical outputs must also be carefully controlled by the author. For instance, Charlene and Harold are both terms in the symbol *person*, but preventing a expansion rule on just *person* from picking Charlene for both instances is not as straightforward. Namely, Charlene could become the subject of Charlene's desire to win the heart of Charlene. Charlene could also become their own rival. Preventing nonsensical redundancies in this CFG approach would require separate grammar symbols for each actor in the story as opposed to having a program generating the narrative while automatically excluding generally undesirable behavior such as this.

In our use of CFGs, the majority of the responsibility of ensuring consistency in the expanded narrative format still belongs to the author. We draw a distinction between two types of variety generation mechanisms that are utilized in our system, namely automated variety and manual variety generation. The automated variety generation is performed through the use of Talk of the Town in terms of different characters, traits, occupations, actions, etc. Glaive also contributes towards this by returning more than one solution to a given problem. The manual variety generation is done through the use of CFGs by authoring different sentence structures based on character traits and by constructing varied world domains in Glaive.

```
(travel indiana usa tanis)
(excavate indiana ark tanis)
(travel indiana tanis usa)
(non-executed (give indiana ark army usa))
(travel nazis tanis usa)
(take nazis ark indiana usa)
(open-ark nazis)
(take army ark nazis usa))
```

Listing 5: A solution to an example problem provided with Glaive.

4.2 Glaive Output

Listing 5 shows a solution to an example planning problem included with Glaive. The problem is based on the film *Indiana Jones and the Raiders of the Lost Ark* and depicts the struggle of multiple actors (US Army, Indiana Jones, Nazis) all competing for possession of the titular Ark of the Covenant. Each actor has intended desired states defined in the problem definition which govern how each actor behaves in the plan solution. The goal states for this problem were defined as *(at army usa), (has army ark)* and *(not (alive nazis))* along with various intentions such as *(intends indiana (alive indiana))* and *(intends nazis (open ark))*.

A particularly interesting aspect of Glaive is that it also mentions when intended actions are omitted from a problem solution as shown by the predicate $(non - executed(...))$. This feature of highlighting non-executed actions illustrates particular points of conflict within a story plan which can be expanded further when translating the story plan to a narrative.

It is important to remember however, that Glaive only returns a solution to a narrative planning problem as a sequence of action steps rather than a structurally complete narrative. Glaive's approach to planning-based narrative generation allows authors to focus more on the types of actions that they would like to see actors perform in a story. These actions are defined separately from an instance of a problem and can be generalized to numerous problem instances matching that domain. Hence, there is an innately high degree of modularity of the author's effort spent in defining the world domain. Given that generated narratives strictly follow the preconditions, effects, and set of actions defined in the world domain, causal relationships can be more tightly controlled and generated narratives resulting from problem solutions are guaranteed to fulfill their goal predicates given that the problem is solvable.

However, the variedness of narratives generated through this approach are not as varied as Tracery's. Glaive's narrative content is determined by constants predefined in a problem definition rather than grammar based expansion rules allowing for random term selection. Even though the simplest type of variable expansion might result in a symbol denoted as $\#food\#$ being referred to as *pizza* or *burger* in some instances, this provides some degree of variety to stories. The planning based approach described here does not replicate this behavior. Additionally, a significant subset of narratives generated for a problem often includes near identical solution steps being executed in different orders but with the same overall effect. This is because the goal-satisfaction oriented approach to planning based narrative generation is only concerned with satisfying goal conditions.

Hence, Glaive's planning based approach to narrative generation allows an author to define a world domain in a very modular fashion with strict control over the types of actions that actors might engage in while supporting actor intentionality but the approach is generally not as "expressive" as a CFG approach.

4.3 Talk of the Town Output

By itself, Talk of the Town generates content for narratives rather than complete narratives. The simulation simulates people living in a fictional town over generations and keeps track of the events that take place during the simulation. As an extensive simulation of a fictional town filled with people with various and detailed personalities, occupations, romantic interests, and motivations, Talk of the Town by itself provides a slew of information that can potentially be used in narratives. Of the three components of our system described here, Talk of the Town has the most variety in its output. This is why it is a powerful tool for generating content for narratives.

When combined with a CFG, Talk of the Town can be a useful way to automatically generate text, dialogue, and narrative content without placing a significant burden on the author. The creators of Talk of the Town have demonstrated this by integrating Talk of the Town with a PCFG designed for generating character dialogue through a system called *Expressionist* [8]. Although our PCFG is similar to the one for *Expressionist* in that it assigns probabilities to production rules, ours differs in that it uses a character's personality attributes from Talk of the Town as parameters passed to a function that determines the weighted probabilities of a set of production rules.

4.4 Stories of the Town Output

Listings 6-8 showcase the results from the three distinct components of our system. Listing 6 shows a problem definition generated by our system by utilizing character information from Talk of the Town. All aspects of characters relevant to the types of stories we might want to generate are represented as predicates that describe the qualities of a character, a character's relationship with other characters, or even the locations of items within the narrative world[3]. Deciding which information to extract and reinterpret as predicates is primarily determined by the preconditions and effects of actions defined in the world domain. Extraneous information will not be used by the Glaive planner since it will have no interpretation in the world domain and is subsequently omitted from the initial problem definition to keep things concise. The narrative goal for the problem in Listing 6 is for Raymond to love Jeanette as shown by the predicate *(loves Raymond Jeanette)*.

Listing 7 shows the set of discrete actions that will achieve the goal defined in the initial world domain mentioned above. We will now explain how this is a valid solution. Assuming that *(loves A B)* is not already true, *(loves A B)* can only be achieved by *A* flirting with *B*—namely *(flirt A B)*. However, a character can only flirt with another character if that character likes them, does not dislike them, and the initiating character is an extrovert. In our predicate logic this means that three preconditions of *(flirt A B)* are *(likes B A)*, ¬*(dislikes B A)*, and *(extrovert A)*. A character can get a character to like them (and subsequently no longer dislike them) if they give them an item that they are in possession of—namely two of the effects of *(give A B item)* are *(likes B A)* and ¬*(dislikes B A)*. Characters gain possession of items by retrieving an item at a location which is represented by *(retrieve A item place)*. It is important to note however, that not all characters are extroverts by default. To address this problem, a character can become an extrovert by executing the *dance* action when they are in possession of an item called the *dance moves brochure* while also being at the *dance studio* location. It should also be noted that whenever a character retrieves an item or interacts with another character, they must be in the same location. Even though this was not described in the brief action descriptions above, it is true in our world domain definition.

With this solution description in mind, the solution can be interpreted as follows: 1. Jeanette went to the Dance Studio in order to retrieve the dance brochure. 2. Jeanette retrieved the dance brochure. 3. Jeanette danced in order to become an extrovert. 4. Jeanette traveled to the park to retrieve a flower. 5. Jeanette retrieved a flower. 6. Jeanette gave a flower to Raymond who was also at the park in order to get Raymond to like them. 7. Now that Jeanette is an extrovert and Raymond now likes Jeanette, Jeanette flirted with Raymond in order to get Raymond to love them.

Lastly, Listing 8 shows one variation of the solution from Listing 7 after being expanded by our PCFG. Each action is iteratively and sequentially parsed by our PCFG which maps each specific action to a series of potential expansion rules. The expansion rules that get applied to an action interpretation directly depend on the personality of the character initiating that action. For instance, the problem solution in Listing 7 lists the action *dance* for Jeanette at the third timestep. How a character feels about executing the *dance* action depends on two specific personality attributes from Talk of the Town: a character's openness to experience and their extroversion. Both of these attributes are binary values which we use to assign probabilities to each potential expansion rule. A character who is both open to experience and an extrovert is assigned a higher probability of eagerly attempting the *dance* action than the alternative expansion rules which can be informally described as deciding to perform the same action with a sense of curiosity and reluctantly performing the action. In order to add an extra layer of complexity to our generated stories, we also use a similar process to describe how someone other than the initiating agent of an action might interpret that action. Namely, the *give* action requires two actors—the giver and the receiver—and we apply our PCFG to both how the giver goes about the *give* action and how the receiver "feels about" receiving a gift. The PCFG when applied to the receiver also uses their personality to influence how they might interpret being on the receiving end of some action. In summation, Listing 8 is an example of one of several different stories we can generate via Stories of the Town which combines CFGs, simulation, and planning-based approaches to narrative generation in a single, cohesive system [4].

```
(:init
   (at Jeanette Sunset-Bistro)
   (love_interest Jeanette Raymond)
   (introvert Jeanette)
   (at Raymond_Follin Starlight-Park)
   (love_interest Raymond Jeanette)
   (extrovert Raymond)
   ...
   (item_at flower Starlight-Park)
   (item_at dance_brochure Dance-Studio)
   (intends Jeanette (loves Raymond Jeanette))
   (intends Raymond (loves Jeanette Raymond)))
(:goal
   (loves Raymond Jeanette))
```

Listing 6: Initial problem formulation for a new example. The format is similar to that of Listing 1.

```
(travel Jeanette Bistro Dance-Studio)
(retrieve Jeanette dance_brochure Dance-Studio)
(dance Jeanette)
(travel Jeanette Dance-Studio Park)
(retrieve Jeanette flower Park)
(give Jeanette Raymond flower Park)
(flirt Jeanette Raymond Park)
```

Listing 7: A problem solution to the initial problem definition in Listing 6.

```
Jeanette Ruff went to Hip Hop Dance Studio from
Sunset Bistro. Jeanette Ruff picked up a dance
moves brochure. Jeanette Ruff figured that the
best way to boost their confidence would be try
out some of the dance moves in the dance moves
brochure. Jeanette Ruff spent several minutes
learning the dance moves to the best of their
```

[3] Only predicates relevant to the characters mentioned in the goal state are included here. The rest have been omitted for clarity.

[4] Examples of more generated stories are listed in the appendix

```
ability. Jeanette Ruff is now brimming with
confidence! Jeanette Ruff traveled from Hip Hop
Dance Studio to Starlight Park. Jeanette Ruff
picked up a flower from the ground. Jeanette
Ruff cheerfully greeted Raymond Follin.
Jeanette Ruff gave Raymond Follin the flower
they obtained. Raymond Follin happily accepted
Jeanette Ruff's gift. Raymond Follin now likes
Jeanette Ruff! Jeanette Ruff noticed that
Raymond Follin was at Starlight-Park as well.
Jeanette Ruff mustered their newfound
confidence to try to flirt with Raymond Follin
to the best of their ability. Raymond Follin
was really impressed by Jeanette Ruff's air of
confidence. Raymond Follin is now smitten with
Jeanette Ruff!
```

Listing 8: A story generated from applying our PCFG to the problem solution from Listing 7.

5 DISCUSSION

We have demonstrated how our system, Stories of the Town, combines three distinct approaches to narrative generation in order to produce narratives that strike a balance between character autonomy and story coherency while using procedurally generated content for its narratives. Although the current state of the system requires careful manipulation of each component, it is able to successfully exploit the strengths of each approach.

The context-free grammar component of Stories of the Town allows for structured grammar symbol expansion to create natural language sentences over which we have a significant degree of control in terms of structure and style. Furthermore, the probabilistic part of our grammar allows us to incorporate a character's personality into our grammar as a way to make the narrative descriptions of characters' actions more consistent with their individual personalities. Our approach to grammars is also significantly more modular than using a CFG to generate an entire story since we map each potential action that a character may take to a series of specialized, potential grammar expansion rules. Each action's grammar expansion rules are defined separately from each other thereby allowing authors to focus on how they want individual actions to be handled by the CFG rather than structuring the CFG to encompass an entire story and its corresponding complexity.

The planning component of Stories of the Town produces plan solutions to planning problems crafted from the personalities, motivations, and relationships of characters simulated in Talk of the Town. These planning problems advocate each character's autonomy by defining intentions and goals specific to their individual interests and are supplemented by a world domain which provides them with the means (i.e. actions) to pursue these intentions and goals. Naturally, the planning component includes the simulation based component which provides the narrative content for the planning component to parse. The planning component uses Talk of the Town to allow it to produce planning problems and solutions that are varied, considerate of individual character details, and coherent. In its current state however, our system masks some of the complexity of Talk of the Town. This is evidenced by the fact that our system does not currently account for events and event sequences that occur during the simulation. We envision further development of the system involving the integration of significant events in Talk of the Town such that they can be used as main narrative points for the planner. With this in mind, we are particularly interested in expanding Stories of the Town to generate narratives that summarize and or reinterpret key events in simulations as an alternative way to understand particular moments and observations that arise through simulations.

Normally, a traditional planning approach to narrative generation might suffer from a lack of varied output but our simulation component is designed to alleviate this drawback. Hence, we are able to produce planning problem solutions with varied output that are still consistent with character personalities and relationships. It should be noted that this approach is limited by the variety of actions one is able to successfully define for the world domain and by being able to extract sufficient information to use for planning problems. However, we can also use the narrative planning component of our system as a means towards measuring the diversity of the types of stories we can generate. This can be achieved by measuring the plan diversity of our plan sets using a domain-independent implementation of normalized compression distance for a plan-plan distance measure for plan diversity as described by Goldman and Kuter [5]. This can provide us with a way to measure the expressiveness of our planning component which is complemented by the perhaps more easily understood expressiveness of our other components.

In conclusion, this paper provides an initial presentation of the Stories of the Town system which we have made publicly available[5]. This system has potential to automatically generate varied, complex, consistent, and goal-oriented narratives. Further development of the system will allow for more integrated end-to-end narrative generation systems that take advantage of the varied representations necessary for such systems.

REFERENCES

[1] John B Black and Robert Wilensky. 1979. An evaluation of story grammars. *Cognitive science* 3, 3 (1979), 213–229.

[2] Hsueh-Min Chang and Von-Wun Soo. 2009. Planning-based narrative generation in simulated game universes. *IEEE Transactions on Computational Intelligence and AI in Games* 1, 3 (2009), 200–213.

[3] Kate Compton, Benjamin Filstrup, and others. 2014. Tracery: Approachable story grammar authoring for casual users. In *Seventh Intelligent Narrative Technologies Workshop*.

[4] Kate Compton, Ben Kybartas, and Michael Mateas. 2015. Tracery: an author-focused generative text tool. In *International Conference on Interactive Digital Storytelling*. Springer, 154–161.

[5] Robert P Goldman and Ugur Kuter. 2015. Measuring plan diversity: Pathologies in existing approaches and a new plan distance metric. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

[6] Mark O Riedl and Robert Michael Young. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research* 39 (2010), 217–268.

[7] James Ryan, Michael Mateas, and Noah Wardrip-Fruin. 2016. Characters who speak their minds: Dialogue generation in Talk of the Town. In *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*.

[8] James Ryan, Ethan Seither, Michael Mateas, and Noah Wardrip-Fruin. 2016. Expressionist: An authoring tool for in-game text generation. In *International Conference on Interactive Digital Storytelling*. Springer, 221–233.

[9] James Owen Ryan, Adam Summerville, Michael Mateas, and Noah Wardrip-Fruin. 2015. Toward characters who observe, tell, misremember, and lie. In *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference*.

[10] Noor Shaker, Julian Togelius, and Mark J Nelson. 2016. *Procedural content generation in games*. Springer.

[5] https://github.com/chris-wayne-miller/storiesofthetown

[11] Stephen G Ware and R Michael Young. 2010. Modeling narrative conflict to generate interesting stories. In *Sixth Artificial Intelligence and Interactive Digital Entertainment Conference*.
[12] Stephen G Ware and R Michael Young. 2014. Glaive: a state-space narrative planner supporting intentionality and conflict. In *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

A APPENDIX

```
Jeanette Ruff traveled from Sunset Bistro to Hip
Hop Dance Studio. A dance moves brochure was
lying on the ground. Jeanette Ruff picked it up.
Jeanette Ruff took a peek at the dance moves in
the dance moves brochure. Jeanette Ruff was a
little embarassed at the idea of practing the
dance moves but figured it was worth a shot.
After practing the moves, Jeanette Ruff feels
more confident! Jeanette Ruff traveled from Hip
Hop Dance Studio to Starlight Park. Jeanette Ruff
picked up a flower. Jeanette Ruff cheerfully
greeted Raymond Follin. Jeanette Ruff gave Raymond
Follin the flower they obtained. Raymond Follin
happily accepted Jeanette Ruff's gift. Raymond
Follin now likes Jeanette Ruff! At Starlight-Park,
Jeanette Ruff encountered Raymond Follin. Seizing
the opportunity, Jeanette Ruff decided to use
their renewed sense of confidence to flirt with
Raymond Follin. Raymond Follin was rather
impressed by Jeanette Ruff's attempt at flirting.
Raymond Follin likes Jeanette Ruff a lot more
now...
```

Listing 9: Narrative variant of Listing 7

```
Russell Pask picked up a panini. Russell Pask
shyly gave their panini to Ronald Pask. Ronald
Pask was a tad surprised but offered to accept
Russell Pask's sudden gift. Ronald Pask feels
like they know Russell Pask a little better
now...
```

Listing 10: Sample narrative output

```
Earnest Egle picked up a dance moves brochure.
Earnest Egle took a peek at the dance moves in the
dance moves brochure. Earnest Egle was a little
embarassed at the idea of practing the dance moves
but figured it was worth a shot. After practing
the moves, Earnest Egle feels more confident! At
Hip-Hop-Dance-Studio, Earnest Egle encountered
Janet Alsbury. Seizing the opportunity, Earnest
Egle decided to use their renewed sense of
confidence to flirt with Janet Alsbury. Janet
Alsbury was rather impressed by Earnest Egle's
attempt at flirting. Janet Alsbury likes Earnest
Egle a lot more now....
```

Listing 11: Sample narrative output

```
Eva O'Neill was in Starlight Park and went to Hip
Hop Dance Studio. A sick dance moves brochure was
lying on the ground. Eva O'Neill picked it up. Eva
O'Neill figured that the best way to boost their
confidence would be try out some of the dance
moves in the dance moves brochure. Eva O'Neill
spent several minutes learning the dance moves to
the best of their ability. Eva O'Neill is now
brimming with confidence! Eva O'Neill traveled
from Hip Hop Dance Studio to Starlight Park. Eva
O'Neill picked up a flower from the ground. Eva
O'Neill shyly gave their flower to Leslie Croker.
Leslie Croker was a tad surprised but offered to
accept Eva O'Neill's sudden gift. Leslie Croker
feels like they know Eva O'Neill a little better
now...At Starlight-Park, Eva O'Neill encountered
Leslie Croker. Seizing the opportunity, Eva
O'Neill decided to use their renewed sense of
confidence to flirt with Leslie Croker. Leslie
Croker was really impressed by Eva O'Neill's air
of confidence. Leslie Croker is now smitten with
Eva O'Neill!
```

Listing 12: Sample narrative output