Solution Density and Search Strategy in Narrative Generation

Cory Siler and Stephen G. Ware, *Member, IEEE*

***Abstract*—Intelligent interactive narrative systems coordinate a cast of non-player characters to make the overall story experience meaningful for the player. Narrative generation involves a trade-off between plot-structure requirements and quality of character behavior, as well as computational efficiency. We study this tradeoff using the example of benchmark problems for narrative planning algorithms. A typical narrative planning problem calls for a sequence of actions that leads to an overall plot goal being met, while also requiring each action to respect constraints that create the appearance of character autonomy. We consider simplified solution definitions that enforce only plot requirements or only character requirements, and we measure how often each of these definitions leads to a solution that happens to meet *both* types of requirements—i.e., the *density* with which narrative plans occur among plot- or character-requirement-satisfying sequences. We then investigate whether solution densities can guide the selection of narrative planning algorithms. We compare the performance of two search strategies: one that satisfies plot requirements first and checks character requirements afterward, and one that continuously verifies character requirements. Our results show that comparing solution densities does not by itself predict which of these search strategies will be more efficient in terms of search nodes visited, suggesting that other important factors exist. We discuss what some of these factors could be. Our work opens further investigation into characterizing narrative planning algorithms and how they interact with specific domains. The results also highlight the diversity and difficulty of solving narrative planning problems.**

I. Introduction

WHEN it comes to developing games with dynamic storylines and elaborate character interactions, manual content authoring techniques such as dialogue trees dominate the industry [1], but intelligent narrative generation technologies could open new opportunities. Rather than requiring steep growth of authoring effort as the length and branching factor of the narratives increase [2], these technologies could allow scale, complexity, and variety to be added easily; rather than restricting player options to match preconceived paths, these games could offer players greater freedom with the narrative adapting at runtime to unexpected player decisions.

Although surprising outcomes from an unbounded "emergent narrative" [3] are sometimes part of a system's appeal, usually the system author will have some requirements for the overall progression of events. For instance, in an educational game or training simulation, the scenario should unfold in a direction that supports the system's pedagogical goals; in a role-playing game, the author may want certain important plot points to occur even if the events in-between can vary. At the same time, the story's believability to the player can benefit from the individual virtual characters behaving like autonomous agents—e.g., acting based on limited information to further their own goals [4]—whether the underlying architecture is a true multiagent system or the characters are being controlled by a hidden central coordinator.

The most appropriate choice of architecture for a narrative generation system depends on the relative importance and difficulty of meeting criteria such as these. We characterize the problem of combining story-structure constraints with character constraints as one of finding members of narrower solution spaces within wider solution spaces, where "solutions" are generated stories that have a given set of desired properties.

Consider the process of designing a narrative generator starting from a fully centralized architecture. With direct control over all of the characters, the central agent has full power over the story structure formed by the collective sum of the characters' actions. However, manipulating the characters freely can lead to a jarring player experience if individual characters show no consistency of their own. The designer can counteract this by imposing some form of character model and requiring the central agent to choose manipulations that are justified according to that model, creating the illusion of autonomy [5]. The architecture starts from a solution space that satisfies story-structure constraints, and narrows it to a solution space that also satisfies character constraints.

Conversely, consider designing a narrative generator starting from a fully multiagent architecture. There is no need to impose the *illusion* of character autonomy because the characters are already autonomous by the nature of the system. However, to ensure that a story with a desired structure emerges, the designer must add additional features—e.g., behind-the-scenes collusion between the character agents [6], partial guidance from an experience manager agent [7], or simply the ability to simulate many possible story trajectories offline and discard them until a satisfying one is found [8]. The architecture starts from a solution space that satisfies character constraints, and narrows it to a solution space that also satisfies story-structure constraints.

This paper extends our previous work [9] providing the first direct investigation of solution spaces in the narrative planning model of narrative generation. A solution to a narrative planning problem is a sequence of character actions that satisfies certain author-oriented and character-oriented requirements.

Manuscript received February 4, 2022. This material is based upon work supported by the National Science Foundation under Grant No. IIS-1911053. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

C. Siler (cory.siler@uky.edu) and S.G. Ware (sgware@cs.uky.edu) are with the Narrative Intelligence Lab, University of Kentucky, Lexington, Kentucky 40508 USA.

We consider relaxed solution definitions consisting of author-oriented or character-oriented requirements. For a variety of benchmark narrative planning problems, we count the solutions that exist for each of the relaxed definitions, as well as for the strict definition incorporating both types of requirements. By doing so, we can see how frequently a strict solution occurs among relaxed solutions, i.e., the density of the former space within the latter ones.

This analysis highlights many examples of cases where a system designed only around author-oriented requirements will often produce results that may be unsatisfactory from a character-oriented perspective, and vice versa, supporting the value of approaches that explicitly model both types of requirements. In addition to being interested in this information for its own sake, we ultimately wish to use it to help guide the design of narrative planning algorithms.

One relevant design factor is computational efficiency. Solution density has previously been used to characterize the performance of search algorithms on classic sequential decision-making problems [10]. We present an experiment that applies it similarly for narrative planning. We test two variants of depth-first search—one prioritizing author-oriented requirements, and one prioritizing character-oriented requirements—to investigate whether the choice of search strategy affects performance in terms of how many search nodes are enumerated and whether solution density can be used to predict which search strategy will enumerate fewer search nodes.

Our results show a significant difference between the two search strategies for some benchmarks, but not others. Among significant results, the more space-efficient search strategy differs over domains and does not consistently align with the solution densities we measured; that is, knowing the solution densities alone is not enough to predict which search strategy will visit fewer nodes for a previously-unknown domain. We conclude that while solution densities offer one tool for comparing domains, they do not by themselves give enough information about a search space to guide algorithm selection, and more analysis tools are needed to complement them.

The rest of this paper is as follows. In Section II, we situate our research within the literature. In Section III, we introduce notation and definitions for the types of narrative generation problems we will be investigating. In Section IV, we enumerate the solutions for a set of narrative generation problems according to different definitions of a solution. In Section V, we define an explanation-first search algorithm and an author-goal-first search algorithm as a component for the experiments that follow. In Section VI, we compare the number of nodes visited by these algorithms, and we relate our findings to the data from Section IV. In Section VII, we discuss the future directions suggested by our results.

II. Related Work

Narrative generation technologies have a wide variety of subcategories and approaches, as surveyed by Kybartas and Bidarra [11]. Outside our scope are approaches that focus on controlling the presentation (discourse) of a narrative, including natural language generation, as well as approaches that focus on generating the setting of a story world or the objects within. Instead, we focus on the generation of plot as a sequence of events within the story world.

Resolving the tension between a system's control over story structure and other desiderata like character believability (as well as player agency [12]) has been an enduring theme in the past two decades of narrative intelligence research. Mateas [13] proposes that narrative systems exist on a spectrum from *strong story*, where a central agent makes all character decisions, to *strong autonomy*, where characters can act with full independence. The advantage of the former is tight control over the overall story structure; the advantage of the latter is the ability to harness the wealth of available work in creating believable autonomous agents. One of the most famous achievements in intelligent interactive narrative, *Façade* [14], strives to balance these advantages using a rich library of local character behaviors, an experience manager that directs the characters with plot-advancing goals, and the ability for a group of characters to perform a joint behavior that is neither centrally imposed nor based fully on individual autonomy. Contrast this with two prominent works from the history of non-interactive story generation: The strongly autonomous TALE-SPIN [15] lets a story emerge based on how characters' motivations, beliefs, and traits interact with the system's physical and social dynamics, while AUTHOR [16] exemplifies strong story by aiming to simulate the mind of an author developing a plot structure rather than simulating the story world itself.

The *narrative planning* family of narrative generation approaches adopts classical planning's language of propositional world state, propositional goals, and declarative action definitions. A plan is taken to be the overall plot structure, with the story characters carrying out the actions that comprise it. The goal-directed nature of planning is appealing because it allows for a form of authorial intent to be enforced; the author can specify desired properties of a state where the plot should eventually lead, i.e., an author goal [17]. It is typically assumed that narrative planning is a tool of a strong-story experience manager that performs all reasoning centrally, although Teutenberg and Porteous [18] provide an example of a partially distributed algorithm where character agents propose candidate actions to the experience manager.

The narrative planner IPOCL [5] follows the paradigm of partial-order planning, which constructs a plan as a directed acyclic graph of actions backwards from the goal, adding actions and orderings only as needed to ensure preconditions and goal propositions will be fulfilled; this guarantees that all plot events are relevant to the outcome of the story by way of fulfilling or enabling fulfillment of the author goal. IPOCL also promotes character believability by introducing character goals, separate from the author goal, and requiring all character actions to be linked to character goal achievement in the same way that all actions overall are linked to author goal achievement.

Later narrative planners iterate on this approach: CPOCL [19] refines the IPOCL algorithm so character actions can be linked to character goal achievement through unexecuted steps, allowing characters' attempts at accomplishing their goals to

fail in the executed plan (e.g., because another character's actions interfered). Glaive [20] keeps CPOCL's concept of explaining character actions with unexecuted steps but divorces it from partial-order planning, instead achieving performance gains through a state-space heuristic search. Sabre [21] performs state-space search to find plans and character action explanations like Glaive but adds a character belief model, including the ability to anticipate actions by other characters. Sabre's model of author goal achievement as a form of story structure, and goal-and-belief-based explainability as a form of character believability, is the basis for our experiments. However, these are not the only notions of story structure and character believability that have been explored in the narrative planning literature; for instance, Shirvani and Ware [22] incorporate an emotional model into character motivation, Porteous et al. [23] use PDDL state trajectory constraints to express mid-narrative authorial intent, and Young [24] and Bae and Young [25] discuss control over story structure to induce suspense and surprise respectively in the audience.

Analyzing a narrative generation domain based on its solution densities can be considered a form of analyzing its *expressive range*, a concept first discussed by Smith and Whitehead [26] in the context of characterizing platform game levels produced by a procedural content generator. Later works apply this concept to narrative generation: Partlan et al. [27] use metrics to evaluate the influence of player choice in branching interactive scenarios, and Kybartas et al. [28, 29] provide tools to visualize the potential for character conflict in possible states of a simulated story world. These forms of analysis, like ours, can help creators of narrative systems understand how design decisions will affect the set of possible stories their systems could produce.

The term "solution density" comes from the heuristic search literature [10], where it refers to the ratio of actual solutions to candidate solutions; for instance, a Boolean satisfiability problem over n variables that has k satisfying assignments is said to have a density of $k/2^n$ [30].

III. Narrative Generation Solution Definitions

In this section, we introduce the notation for the formal model of narrative generation we use in our experiments. While it is unfeasible to define a single formalism that is representative of the entire diverse spectrum of narrative generation approaches, we aim to capture common features and generate solutions for some well-defined existing narrative generation problems. To achieve this, we use a modified version of a formalism introduced by Shirvani et al. [4] and implemented in Sabre [21]. The formalism ultimately derives from STRIPS-style [31] classical planning. We build up to a definitions for strict narrative-planning solutions incorporating author and character constraints, and relaxed solution definitions incorporating only author or only character constraints.

A. Problem Instances

We consider narrative generation problems consisting of a tuple $\langle s_0, V, C, A, g_a, G(c)\rangle$.

V is a set of propositional variables. A state assigns a value of true or false to each of those variables. s_0 is a state called the *initial state*, which is supplied to describe the starting state of the story world.

C is a set of objects representing the story's characters. Characters have (possibly-wrong) beliefs about the state of the world; for every character $c \in C$ and variable $v \in V$, we define an additional variable $believes(c, v) \in V$ representing the proposition that a character c believes v is true (otherwise, c believes v is false). Beliefs can be arbitrarily nested; $believes(c_1, believes(c_2, v))$ is the proposition that character c_1 believes character c_2 believes v is true. A state tracks all characters' beliefs, including beliefs about each other's beliefs and so on. (See Shirvani et al. [32] for a discussion of how the infinite set of resulting beliefs can be finitely represented.) A character's beliefs also correspond to a state; at any time, a character believes exactly one value for each variable.

A is a set of actions. Each action has preconditions and effects, which are propositions over V. An action can be taken only in a state where its preconditions holds, and results in a state where its effects hold. Every action also defines a (possibly empty) set of *consenting characters* from C. Intuitively, consenting characters are the characters that "take" an action; in some solution types defined later, these also affect which actions can be taken.

g_a is the author's goal, a proposition over V. For every character $c \in C$, $G(c)$ is a set of propositions over V; these constitute character goals for c.[1] These are used to define solutions later in this section.

We illustrate with a running example based on the `grandma` domains [33]. The character set C includes the Merchant and the protagonist Tom. The variable set V defines the locations of the characters and of items such as a Coin and a Potion (e.g., the variable $at(Tom, Cottage)$ tracks whether Tom is at the Cottage). The initial state s_0 declares initial facts, such as Tom being at the Cottage (i.e., $at(Tom, Cottage)$ is true) with the Coin, the Merchant being at the Market with the Potion, Tom believing the Merchant is at the Market, and the Merchant not knowing Tom is at the cottage (i.e., $believes(Merchant, at(Tom, Cottage))$ is false). $G(c)$ says the Merchant wants to have the Coin (i.e., $at(Coin, Merchant) \in G(Merchant)$) and that Tom wants to have the Potion at the Cottage. g_a says the author goal is also for Tom to have the Potion at the Cottage (i.e., $g_a = (at(Tom, Cottage) \wedge at(Potion, Tom))$.

Actions in A include *travel*, *give*, *pick-up*, *kill*, and *buy*. The action for Tom to *travel* from the Cottage to the Market, for instance, has the precondition that Tom is at the Cottage ($at(Tom, Cottage)$) and the effects that Tom is at the Market instead and that characters at these locations update their beliefs to reflect this ($at(Tom, Market) \wedge \neg at(Tom, Cottage) \wedge \forall_{c \in C} at(c, Cottage) \rightarrow believes(c, at(Tom, Market)) \wedge \cdots$).

[1] Sabre uses a richer syntax where desires are specified in terms of a utility function instead of propositional goals; however, since the original forms of our domains generally use propositional goals and we set up the utility functions to handle these goals equivalently, we will use a goal-based formalism in this paper for brevity.

B. Explaining Actions

What differentiates instances of the above problem from classical planning instances is the addition of character goals, character beliefs, and consenting characters for actions. This makes possible some of the solution definitions we will consider in Section III-C that impose character-based constraints on chosen actions, reinforcing the idea that story characters should behave like autonomous agents with their own goals and models of the world. These constraints say that when an action is selected to be taken by character c, there should exist a hypothetical plan from c's perspective where the action helps lead to one of c's goals in $G(c)$ being achieved, thus giving c a reason to participate in the action.

Let s be a state. When s holds, let s_{bel} be the state character c believes the world is in, as defined by the $believes$ propositions in s. An action $a \in A$ taken in s is *explained* for consenting character c iff:

1) Starting from state s_{bel} where some goal $g \in G(c)$ does not hold, there exists a legal sequence of actions (called an explanation) that begins with action a and ends in a state where g holds.
2) All actions in the explanation have consenting characters.[2]
3) If an action in the explanation has a consenting character $c' \neq c$, that action is explained for c'.
4) The explanation does not contain a strict subsequence that also meets these criteria.

The first criterion ensures that from c's perspective, a could be part of a plan to achieve a goal; the last criterion ensures that a_i is actually relevant to the goal achievement, rather than being redundant. The other criteria ensure that if c anticipates an action taking place outside of its own direct control, that action is plausible based solely on c's awareness of other characters' beliefs and desires.

Returning to our `grandma` example, suppose we are in the initial state and Tom *travel*s from the Cottage to the Market. To explain this action, we consider a plan Tom could have that begins with this action and achieves his goal: He could use his Coin to *buy* the Potion from the Merchant at the Market, and then *travel* back to the Cottage. The *buy* action can only be included if it is explainable for the Merchant, as she is another consenting character. We confirm that it is explainable, as it immediately achieves the Merchant's goal of having the Coin.

C. Solution Definitions

Starting from the initial state s_0, consider a sequence of actions $\langle a_1, a_2, \cdots, a_n \rangle$ where applying the effects of any action a_i to s_i results in a state s_{i+1}. We call such a sequence *legal* if each a_i's preconditions hold in s_i. Possible "solutions" to a narrative generation problem are taken from among legal action sequences. We consider multiple definitions of a solution; given a narrative generation instance $P = \langle s_0, V, C, A, g_a, G(c) \rangle$, we define:

- $Auth(P)$: The set of *author-constraint-satisfying* sequences, namely, those action sequences $\langle a_1, a_2, \cdots, a_n \rangle$ where the author goal g_a holds precisely after a_n is taken.
- $Char(P)$: The set of *character-constraint-satisfying* sequences, namely, those action sequences where each action is explained for its consenting characters.
- $Auth(P) \cap Char(P)$: The set of sequences that are both author-constraint- and character-constraint-satisfying.

For instance, in `grandma`, an example of an $Auth(P) \cap Char(P)$ solution is the one where Tom *travel*s to the Market, *buy*s the Potion, and *travel*s back to the Cottage. An example of a solution in $Auth(P)$ but not $Char(P)$ is one where the Merchant *travel*s to the Cottage and *give*s Tom the Potion; the Merchant initially has no reason to *travel* to the Cottage as she does not know Tom is there, and she has no reason to *give* him the Potion as it does not help her get the Coin. An example of a solution in $Char(P)$ but not $Auth(P)$ is one where Tom *travel*s to the Market only for the Merchant to *kill* him so she can *pick-up* his Coin; the Merchant acts consistently with her own goal but prevents the author goal from being achieved.

IV. Measuring Solution Densities

In this section, we count solutions to characterize the solution spaces of a variety of narrative generation domains. By "domain", we mean a set of problem instances sharing the same variables, characters, and actions; instances from a domain may have different initial states and goals. We aimed to represent domains from a variety of authors and contexts; for instance, some were designed to gauge player perception of character believability in an interactive adventure game (e.g., [33]), to benchmark (e.g., Ware and Young [20]) or illustrate key features of a planner (e.g., Christensen et al. [34]), or to serve as part of a larger story generation pipeline (e.g., Cardona-Rivera and Li [35]).

We used the following domains adapted for the Sabre narrative planner [21].[3]

- `blackbeard`, based on the Treasure Island domain from Shirvani et al. [32].
- `fantasy` from Ware and Young [36], adapted to add character beliefs.
- `grandma-lose`, the domain used for the simple adventure game in the experiments of Ware et al. [33], but with the author goal of the protagonist dying.
- `grandma-win`, the same domain except with the author goal matching the protagonist goal. (The original domain has the disjunction of the "win" and "lose" goals as the author goal.)
- `hospital` from Porteous et al. [37], adapted to add character beliefs and intentions and to have a solution space small enough to fully enumerate.
- `hubris` from Christensen et al. [34].
- `lovers` from Farrell and Ware [38].
- `raiders` from Ware and Young [20], adapted to add character beliefs.

[2] In other words, a character can anticipate how other characters will behave, but has no means of predicting events beyond any character's control.

[3] Sabre and full domain definitions will be available in the near future at: http://cs.uky.edu/~sgware/projects/sabre/

- `red` from the Porteous et al. [37] adaptation of Riedl [17], adapted to add character beliefs and intentions.
- `space` from Ware and Young [36], adapted to add character beliefs.
- `villains`, a domain used for Cardona-Rivera and Li's PlotShot [35], omitting "discourse" actions used to determine presentation rather than plot events.

Each of these domains is paired with a single "canonical" problem instance, except `hospital` which comes with ten manually-authored instances and `lovers` for which we used ten randomly-generated instances known to be solvable.

In order to enumerate the action sequences in $Auth(P)$, $Char(P)$, and $Auth(P) \cap Char(P)$ for each problem instance P, we needed to limit the size of the solution spaces, as many instances would allow an infinite number of solutions by repeating cycles of actions. For each problem instance, we imposed a maximum *depth* on any given solution. Maximum depth d means the solution can only be d actions long. Depth limit also affects explanation length: If an action appears at depth i and an explanation of length j is needed to explain it, we require $i+j \leq d$. We generated and counted all solutions up to a maximum depth for each solution definition. We considered only the executed sequence for determining what counts as a unique solution; if there were multiple ways to explain the same executed sequence, it still contributed only one to the solution count. Furthermore, since many solutions in $Char(P)$ are simply truncations of longer solutions, we considered only maximal sequences with respect to the solution definition and maximum depth, i.e., sequences that were extended as far as they could go within the depth limit while still meeting the solution definition.

Tables I shows the number of solutions for each domain with each solution definition. For multi-instance domains, the values shown are averaged over the instances.

The table also shows as a ratio how densely $Auth(P) \cap Char(P)$ solutions occur among overall $Auth(P)$ or $Char(P)$, calculated by dividing the solution count for $Auth(P) \cap Char(P)$ by the solution count for the other space. In other words, this solution density expresses the probability that a randomly-sampled sequence in $Auth(P)$ or $Char(P)$ will be an $Auth(P) \cap Char(P)$ solution. Note that by this metric, a larger $Auth(P)$ or $Char(P)$ does not inherently mean rarer $Auth(P) \cap Char(P)$ solutions; what matters is the proportional size of the latter space. Note also that solution density does not directly express the probability that an $Auth(P)$ or $Char(P)$ sequence *produced by a narrative generation algorithm* will be an $Auth(P) \cap Char(P)$ solution, since there are nonrandom elements to these algorithms; however, our experiments in Section VI aim to capture whether the random-sampling probability is a good proxy for the difficulty of algorithmic generation.

By default, we set the maximum depth to be the lowest depth at which solutions for $Auth(P) \cap Char(P)$ appear; we make reference to these results for our Section VI experiments, since those experiments revolve around an iterative-deepening search that stops at the first depth where it finds $Auth(P) \cap Char(P)$ solutions. However, for some single-instance domains (limited by available computation time and space), we repeated the data collection at larger maximum depths; these additional results are not referenced in the later experiments but are presented for their own sake in the table. For instances with several points of data, these results are also illustrated as plots in Figure 1.

V. Search Strategies

A typical narrative planner searches for a solution defined much like our $Auth(P) \cap Char(P)$ space. It generates an author-goal-achieving action sequence to be executed, plus unexecuted sequences constituting explanations, i.e., proofs that the actions meet the character constraints. Sabre [21] does this in what we will call an *explanation-first* search: Before it adds an action to the tentative executed sequence, it generates an explanation for that action, discarding the action if no explanation is found. It assembles an author-goal-achieving sequence from already-explained actions, so that when the final author-goal-achieving action is added to the sequence, the whole sequence is known to be a valid narrative plan. We will refer to this search process as ExpFirst, and we detail it in Algorithm 1. Generating explanations is itself a search process; the explanation process, Explain in Algorithm 2, likewise proceeds in an explanation-first manner, but it treats a character's beliefs as true and tries to satisfy character goals rather than taking the actual world state and satisfying author goals.

Algorithm 1 ExpFirst$(s, A, g_a, G(c), d, \pi)$

Input: Initial state s_0, action set A, author goal g_a, character goals $G(c)$, depth limit d, current plan π (the empty plan in the initial call)

Output: A sequence of actions in $Auth(P) \cap Char(P)$, or null if none exists within the depth limit

```
s ← the state resulting from applying π to s_0
if s ⇒ g_a then
    return π
else if d = 0 then
    return null
Nondeterministically choose action a ∈ A legal in s
for all c_a that are consenting characters for a do
    if ¬Explain(s, A, G(c), d − 1, ⟨a⟩) then
        return null
π′ ← the plan resulting from appending a to π
return ExpFirst(s_0, A, G(c), d − 1, π′, c_a)
```

In contrast, some narrative planners (e.g., Glaive [20]) search for solutions in an *author-goal-first* fashion: First they find a sequence of not-yet-explained actions that achieves the author goal (effectively, a classical plan), and then they try to generate an explanation for each of those actions retroactively. We stereotype this process as AuthFirst in Algorithm 3. It differs from ExpFirst only in the timing of explanation generation within the search for an author-goal-achieving state; to focus on the effects of this difference, we keep the explanation procedure (Explain) the same between both search strategies.

We use the vocabulary of nondeterministic choice for brevity; in practice, we implement action choice as a depth-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TG.2022.3149529, IEEE Transactions on Games

TABLE I
SOLUTION COUNTS AND DENSITIES

domain	max depth	$Char(P)$	$Auth(P)$	$Auth(P) \cap Char(P)$	$Auth(P) \cap Char(P)$ density within $Char(P)$	$Auth(P) \cap Char(P)$ density within $Auth(P)$
blackbeard	3	6	4	2	0.333	0.500
blackbeard	4	23	26	5	0.217	0.192
blackbeard	5	86	165	19	0.221	0.115
blackbeard	6	299	926	60	0.201	0.065
blackbeard	7	1004	5239	188	0.187	0.036
blackbeard	8	4831	28893	1149	0.238	0.040
blackbeard	9	25986	159453	6973	0.268	0.044
fantasy	6	12	18	12	1.000	0.667
fantasy	7	12	1322	12	1.000	0.009
grandma-lose	4	8	1351	1	0.125	0.001
grandma-lose	5	37	13830	2	0.054	< 0.001
grandma-lose	6	403	140677	26	0.065	< 0.001
grandma-lose	7	4147	1437070	228	0.055	< 0.001
grandma-win	5	39	344	1	0.026	0.003
grandma-win	6	450	4485	11	0.024	0.002
grandma-win	7	4786	54866	129	0.027	0.002
hubris	5	3	12	1	0.333	0.083
hubris	6	3	48	3	1.000	0.063
hubris	7	4	165	4	1.000	0.024
hubris	8	4	528	4	1.000	0.008
raiders	7	10	706	3	0.300	0.004
raiders	8	9	3631	5	0.556	0.001
raiders	9	19	16386	8	0.421	< 0.001
raiders	10	16	77739	12	0.750	< 0.001
raiders	11	35	351553	20	0.571	< 0.001
red	5	515	2	2	0.004	1.000
space	3	4	1	1	0.250	1.000
space	4	9	9	2	0.222	0.222
space	5	20	70	6	0.300	0.086
space	6	68	407	22	0.324	0.054
space	7	162	2095	58	0.358	0.028
space	8	459	10057	161	0.351	0.016
space	9	1545	46732	530	0.343	0.011
space	10	3930	213373	1386	0.353	0.006
villains	8	1348	1476	360	0.267	0.244
hospital (avg.)	4.3	416.2	1504.9	216.0	0.520	0.143
lovers (avg.)	4.4	315.0	1237.5	19.6	0.062	0.016

Algorithm 2 Subroutine EXPLAIN$(s, A, G(c), d, \pi_{exp}, c_{exp})$

Input: Initial state s, action set A, character goals $G(c)$, depth limit d, explanation so far π_{exp}, character c_{exp};

Output: Boolean: Whether the first action of π_{exp} can be explained for c_{exp} in state s by extending π_{exp} with maximum depth depth d

$s_{bel} \leftarrow$ the state believed by c_{exp} in s
$s'_{bel} \leftarrow$ the state resulting from applying π_{exp} to s_{bel}
if s'_{bel} is undefined **then**
  return false
else if π_{exp} nonredundantly achieves a goal for c_{exp} **then**
  return true
else if $d = 0$ **then**
  return false
Nondeterministically choose action $a \in A$ legal in s'_{bel}
for all $c_a \neq c_{exp}$ that are consenting characters for a **do**
  if $\neg$EXPLAIN$(s, A, G(c), d-1, \langle a \rangle)$ **then**
    return false
$\pi' \leftarrow$ the plan resulting from appending a to π_{exp}
$s' \leftarrow$ the state resulting from taking a in s
return EXPLAIN$(s, A, G(c), d-1, \pi', c_{exp})$

Algorithm 3 AUTHFIRST$(s, A, g_a, G(c), d, \pi)$

Input: Initial state s_0, action set A, author goal g_a, character goals $G(c)$, depth limit d, current plan π (the empty plan in the initial call)

Output: A sequence of actions in $Auth(P) \cap Char(P)$, or null if none exists within the depth limit

$s \leftarrow$ the state resulting from applying π to s_0
if $s \Rightarrow g_a$ **then**
  return π
else if $d = 0$ **then**
  return null
Nondeterministically choose action $a \in A$ legal in s
$\pi' \leftarrow$ the plan resulting from appending a to π
$\pi_{full} \leftarrow$ AUTHFIRST$(s_0, A, G(c), d-1, \pi', c_a)$
if $\pi_{full} =$ null **then**
  return null
for all c_a that are consenting characters for a **do**
  if $\neg$EXPLAIN$(s, A, G(c), d-1, \langle a \rangle)$ **then**
    return null
return π_{full}

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TG.2022.3149529, IEEE Transactions on Games

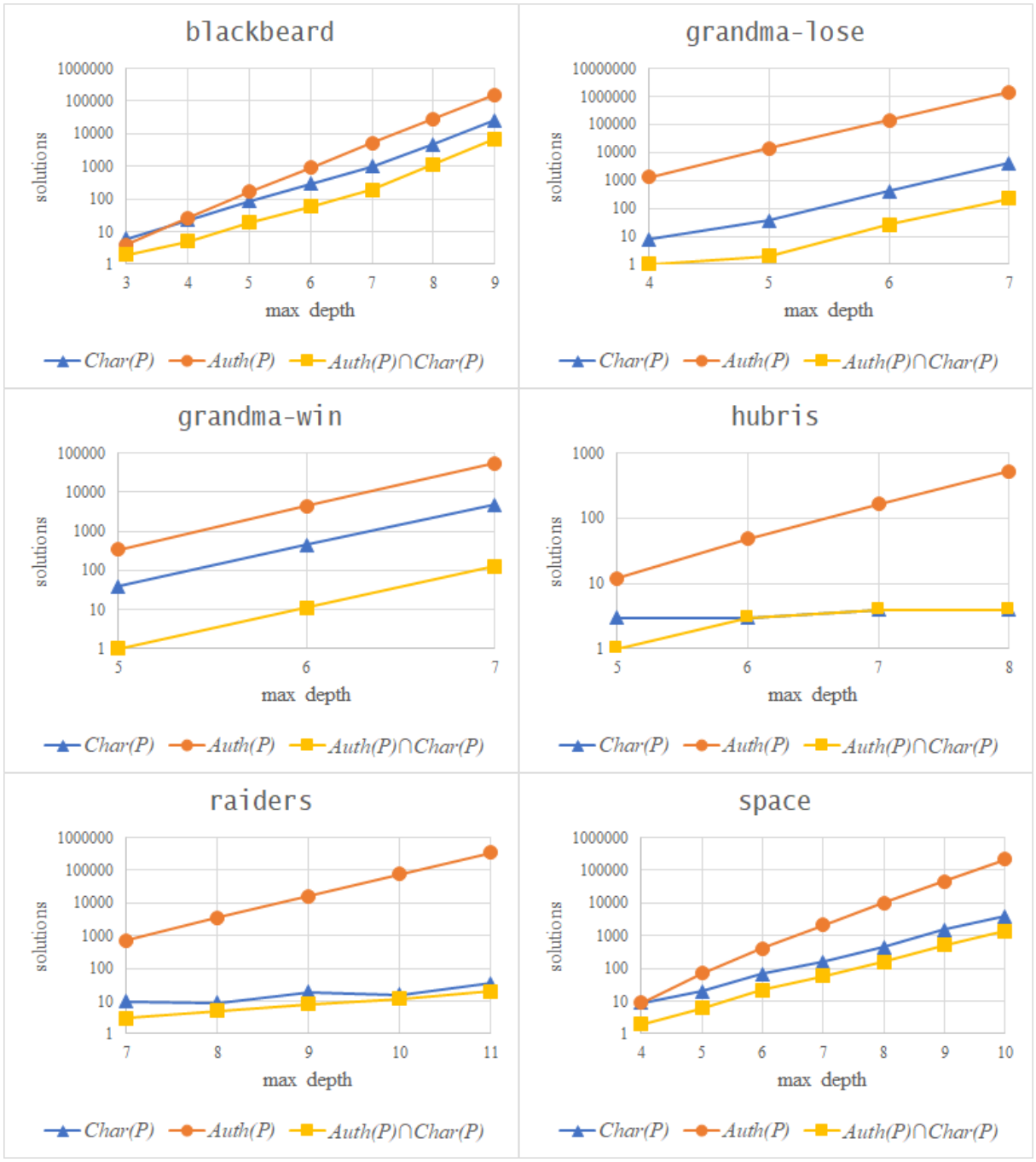

Fig. 1. Visualizations of some of the data from Table I. Note the logarithmic scale.

first search procedure that must spend computational effort trying candidate actions that will not be used in the final plan. ExpFirst risks wasting effort by generating unneeded explanations, because an explained action may not lead to the author goal. In other words, the strategy is efficient at maintaining a sequence known to be in $Char(P)$, but wasteful if the sequence is not also in $Auth(P)$. AuthFirst averts this risk because it does not explain an action until the action is already known to lead to the author goal, but instead it risks wasting effort by generating sequences that turn out not to be explainable. In other words, it prioritizes finding a sequence known to be in $Auth(P)$, but is wasteful if the sequence is not also in $Char(P)$.

VI. Comparing Search Strategies

This section presents our results from benchmarking the search algorithms in Section V on the domains in Section IV in order to investigate how the choice of author- or character-constraint prioritization in the search process relates to the densities of narrative planning ($Auth(P) \cap Char(P)$) solutions within the relaxed author-oriented ($Auth(P)$) or character-oriented ($Char(P)$) solution spaces.

We implemented AuthFirst alongside the existing ExpFirst implementation in the Sabre narrative planner. As

Sabre runs its state-space search, it tracks the *nodes visited* during the search process, i.e., the number of state-action pairs considered, or the total number of recursive calls to AUTHFIRST/EXPFIRST and EXPLAIN. We use nodes-visited count as our metric for comparing the two search strategies; while it does not capture *all* aspects of an algorithm's space and time consumption, it offers a proxy measure that is independent of low-level implementation details (assuming, as is true in our implementation, that candidate actions from a given state are explored in a random order) and the platform on which the experiments are run.

To capture a "fair" search process without preexisting knowledge of the depth at which solutions exist, we ran iterative-deepening versions of each of our strategies, i.e., we started at depth limit $d = 1$, restarted the process with $d = 2$ if no solution was found, and so on, tracking the cumulative nodes-visited count for all successive calls. Note that this means only minimum-depth $Auth(P) \cap Char(P)$ solutions were found—e.g., in reference to Table I, the search found only depth-3 solutions for `blackbeard`, depth-6 for `fantasy`, and so on.

We collected search data for 10 runs each of AUTHFIRST and EXPFIRST for each domain, randomizing the order in which candidate actions were considered at each search node. For single-instance domains, the 10 runs were on the same problem instance; for multiple-instance domains, we used a different instance per run. Table II shows the results: For a domain in the first column, the average nodes-visited count for the AUTHFIRST and EXPFIRST strategies are shown in the second and third columns respectively. The fourth column identifies which of these strategies visited fewer nodes. The fifth column shows the p-value (two-tailed Student's T-test, independent similar-variance samples) for the hypothesis that the nodes-visited count differs between AUTHFIRST and EXPFIRST. Values indicating significance ($p < 0.05$) are bolded.

Four domains showed statistically significant differences between the two search strategies: `grandma-win`, `raiders`, `space`, and `villains`. Among the domains with significant differences, neither search strategy consistently dominated the other over all domains, nor was there a clear relationship between which relaxed solution space had a higher $Auth(P) \cap Char(P)$ solution density (at the first depth where solutions occurred)) and which search strategy required fewer nodes visited; `grandma-win`, `raiders`, and `villains` all had a higher solution density within $Char(P)$, but `grandma-win` had AUTHFIRST visit fewer nodes while the other two had EXPFIRST visit fewer.

This contrasts with a prediction made in our previous work [9]. Recall that AUTHFIRST aims to avoid wasting effort finding explanations that do not end up in the final solution because the explained actions do not lead to an author goal, while EXPFIRST aims to avoid wasting effort finding executed actions that do not end up in the final solution because they are not explainable. This would suggest AUTHFIRST outperforming EXPFIRST as the solution density within $Auth(P)$ becomes higher relative to the solution density within $Char(P)$, and vice versa, which did not consistently hold in our data.

However, this prediction overlooks unmodeled factors that could affect search efficiency. For instance, by emphasizing the number of explanations that the search process needs to generate, it assumes that the amount of effort required per explanation is roughly the same. One way that this assumption could fail to hold is based on our practice of limiting search depth. The earlier an action occurs in the executed sequence, the longer its explanation is allowed to be, creating a larger possible search tree for finding that explanation. All else being the same, trying to explain an unexplainable action toward the beginning of the executed sequence wastes more effort than trying to explain one toward the end. This pattern penalizes EXPFIRST more heavily when the unexplainable actions are concentrated closer to the initial state in state space.

VII. CONCLUSIONS

This paper has taken the concept of solution density analysis from the combinatorial search literature and applied it to narrative generation. Examining solution density can help domain authors examine the range of possible narratives that might result from their domains based on criteria like plot structure and character believability; it can also help narrative planning algorithm designers make design decisions by examining the search spaces their algorithms will need to explore.

At the same time, our experiments show that the relative sizes of author- and character-oriented solution spaces alone are not enough to explain the difference between different narrative planning algorithms' performance. The model Vempaty et al. [10] use to characterize algorithms for traditional search problems considers both solution density and a form of branching factor; the unique requirements of narrative planning mean that explaining and predicting algorithm performance will likely require new tools alongside these classic ones.

We acknowledge the threats to external validity (i.e., generalizability of our findings outside of our specific context) of our experiments; the process we have used to characterize narrative planning domains and algorithms does not precisely reflect how a designer of a real system might do so. Firstly, in order to investigate the relationship between solution densities and search performance, we pre-computed solutions to a set of problems and then ran search algorithms on that same set of problems. In a practical application, if we could precompute the solutions to all instances we would encounter, we would not fast search algorithms to begin with; it is more realistic to analyze only a small subset sampled from many possible instances. Secondly, the algorithms we tested are straightforward variants of depth-first search and do not have the optimizations that would be needed for good performance in a practical application; instead, they are deliberately oversimplified to focus on the effects of immediately explaining character actions versus deferring explanation. Thirdly, since narrative planning has so far not been used in commercial applications and has appeared in only a few complete playable experiences by researchers (such as *The Best Laid Plans* [39]), the domains we used do not necessarily resemble what might be used commercially, although we have assembled a set of domains as diverse as we could from what is currently available.

TABLE II
SEARCH STRATEGY EXPERIMENTAL RESULTS

domain	AUTHFIRST avg. nodes visited	EXPFIRST avg. nodes visited	winner	p
blackbeard	142.1	157.6	AUTHFIRST	0.429
fantasy	3305198.2	3246841.1	EXPFIRST	0.948
grandma-win	80136.4	155164.6	AUTHFIRST	**0.006**
grandma-lose	10085.4	11537.0	AUTHFIRST	0.393
hubris	310.0	321.8	AUTHFIRST	0.694
raiders	37807.7	12773.3	EXPFIRST	**<0.001**
red	73200.1	93720.7	AUTHFIRST	0.245
space	50.3	37.5	EXPFIRST	**0.036**
villains	1081257.6	1753159.9	AUTHFIRST	**<0.001**
hospital	4544.4	6046.1	AUTHFIRST	0.575
lovers	961436.7	288309.8	EXPFIRST	0.410

Despite our experiments' focus on a narrow formulation of narrative generation, some observations are relevant to other areas of narrative intelligence. For instance, why are author-constraint-satisfying sequences are more common than character-constraint-satisfying sequences in most instances? One factor is that the explanations needed to satisfy character constraints constitute standalone narratives in themselves, so a character-constraint-satisfying sequence requires the joint existence of solutions to many subproblems rather than the singular problem solved by an author-constraint-satisfying sequence. While we have taken a perspective of *limiting* a space of generated stories, this recursive feature of narrative can also be used to *broaden* a story space by discovering, transforming, and recombining story fragments [40].

ACKNOWLEDGMENT

We thank Rachelyn Farrell for writing the Sabre versions of several of the narrative planning domains appearing here. We thank the anonymous reviewers for their thoughtful suggestions. We thank the University of Kentucky Center for Computational Sciences and Information Technology Services Research Computing for their support and use of the Lipscomb Compute Cluster and associated research computing resources.

REFERENCES

[1] A. Freed, "Branching conversation systems and the working writer," Gamasutra, 2014.

[2] J. O. Ryan, M. Mateas, and N. Wardrip-Fruin, "A lightweight videogame dialogue manager." in *Joint International Conference of DiGRA and FDG*, 2016.

[3] R. Aylett, "Narrative in virtual environments - towards emergent narrative," in *AAAI Fall Symposium on Narrative Intelligence*, 1999, pp. 83–86.

[4] A. Shirvani, R. Farrell, and S. G. Ware, "Combining intentionality and belief: revisiting believable character plans," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2018, pp. 222–228.

[5] M. O. Riedl and R. M. Young, "Narrative planning: balancing plot and character," *Journal of Artificial Intelligence Research*, vol. 39, no. 1, pp. 217–268, 2010.

[6] M. Mateas and A. Stern, "A Behavior Language: Joint action and behavioral idioms," in *Life-Like Characters*. Springer, 2004, pp. 135–161.

[7] ——, "Towards integrating plot and character for interactive drama," in *Socially Intelligent Agents: Creating Relationships with Humans and Robots*, K. Dautenhahn, A. Bond, L. Cañamero, and B. Edmonds, Eds. Kluwer Academic Publishers, 2002, pp. 221–228.

[8] S. Riegl and T. Veale, "Live, die, evaluate, repeat: Do-over simulation in the generation of coherent episodic stories." in *International Conference on Computational Creativity*, 2018, pp. 80–87.

[9] C. Siler and S. G. Ware, "A good story is one in a million: solution density in narrative generation problems," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2020, pp. 123–129.

[10] N. R. Vempaty, V. Kumar, and R. E. Korf, "Depth-first versus best-first search." in *National Conference on Artificial Intelligence*, 1991, pp. 434–440.

[11] Q. Kybartas and R. Bidarra, "A survey on story generation techniques for authoring computational narratives," *IEEE Transactions on Computational Intelligence and Artificial Intelligence in Games*, vol. 9, no. 3, pp. 239–253, 2016.

[12] M. O. Riedl and V. Bulitko, "Interactive narrative: an intelligent systems approach." *AI Magazine*, vol. 34, no. 1, pp. 67–77, 2013.

[13] M. Mateas, "Interactive drama, art and artificial intelligence," Ph.D. dissertation, Carnegie Mellon University, 2002.

[14] M. Mateas and A. Stern, "Façade: An experiment in building a fully-realized interactive drama," in *Game Developers Conference*, vol. 2, 2003, pp. 4–8.

[15] J. R. Meehan, "TALE-SPIN, an interactive program that writes stories," in *International Joint Conference on Artificial Intelligence*, 1977, pp. 91–98.

[16] N. Dehn, "Story generation after TALE-SPIN," in *International Joint Conference on Artificial Intelligence*, vol. 81, 1981, pp. 16–18.

[17] M. O. Riedl, "Incorporating authorial intent into generative narrative systems," in *AAAI Spring Symposium: Intelligent Narrative Technologies II*, 2009, pp. 91–94.

[18] J. Teutenberg and J. Porteous, "Efficient intent-based narrative generation using multiple planning agents," in *International Conference on Autonomous Agents and*

Multiagent Systems, 2013, pp. 603–610.

[19] S. G. Ware and R. M. Young, "CPOCL: a narrative planner supporting conflict," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2011, pp. 97–102.

[20] ——, "Glaive: a state-space narrative planner supporting intentionality and conflict," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2014, pp. 80–86.

[21] S. G. Ware and C. Siler, "Sabre: A narrative planner supporting intention and deep theory of mind," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2021.

[22] A. Shirvani and S. Ware, "A formalization of emotional planning for strong-story systems," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, no. 1, 2020, pp. 116–122.

[23] J. Porteous, M. Cavazza, and F. Charles, "Applying planning to interactive storytelling: Narrative control using state constraints," *ACM Transactions on Intelligent Systems and Technology*, vol. 1, no. 2, pp. 1–21, 2010.

[24] R. M. Young, "Notes on the use of plan structures in the creation of interactive plot," in *AAAI Fall Symposium on Narrative Intelligence*, 1999, pp. 164–167.

[25] B.-C. Bae and R. M. Young, "A computational model of narrative generation for surprise arousal," *IEEE Transactions on Computational Intelligence and Artificial Intelligence in Games*, vol. 6, no. 2, pp. 131–143, 2014.

[26] G. Smith and J. Whitehead, "Analyzing the expressive range of a level generator," in *Workshop on Procedural Content Generation in Games*. ACM, 2010.

[27] N. Partlan, E. Carstensdottir, S. Snodgrass, E. Kleinman, G. Smith, C. Harteveld, and M. S. El-Nasr, "Exploratory automated analysis of structural features of interactive narrative," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 14, no. 1, 2018.

[28] Q. Kybartas, C. Verbrugge, and J. Lessard, "Tension space anaylsis for emergent narrative," *IEEE Transactions on Games*, 2020.

[29] ——, "A sketch-based tool for authoring and analyzing emergent narratives," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, no. 1, 2020, pp. 319–321.

[30] Y. Hanatani, T. Horiyama, and K. Iwama, "Density condensation of boolean formulas," *Discrete Applied Mathematics*, vol. 154, no. 16, pp. 2263–2270, 2006.

[31] R. E. Fikes and N. J. Nilsson, "STRIPS: a new approach to the application of theorem proving to problem solving," *Artificial Intelligence*, vol. 2, no. 3, pp. 189–208, 1971.

[32] A. Shirvani, S. G. Ware, and R. Farrell, "A possible worlds model of belief for state-space narrative planning," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2017, pp. 101–107.

[33] S. G. Ware, E. T. Garcia, A. Shirvani, and R. Farrell, "Multi-agent narrative experience management as story graph pruning," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2019, pp. 87–93.

[34] M. Christensen, J. Nelson, and R. Cardona-Rivera, "Using domain compilation to add belief to narrative planners," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, no. 1, 2020, pp. 38–44.

[35] R. E. Cardona-Rivera and B. Li, "PlotShot: Generating discourse-constrained stories around photos," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2016.

[36] S. G. Ware and R. M. Young, "Validating a plan-based model of narrative conflict," in *International Conference on Foundations of Digital Games*, 2012, pp. 220–227.

[37] J. Porteous, J. F. Ferreira, A. Lindsay, and M. Cavazza, "Automated narrative planning model extension," *Autonomous Agents and Multi-Agent Systems*, vol. 35, no. 2, pp. 1–29, 2021.

[38] R. Farrell and S. Ware, "Narrative planning for belief and intention recognition," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, no. 1, 2020, pp. 52–58.

[39] S. G. Ware and R. M. Young, "Intentionality and conflict in The Best Laid Plans interactive narrative virtual environment," *IEEE Transactions on Computational Intelligence and Artificial Intelligence in Games*, vol. 8, no. 4, pp. 402–411, 2015.

[40] M. O. Riedl, "Story planning: Creativity through exploration, retrieval, and analogical transformation," *Minds and Machines*, vol. 20, no. 4, pp. 589–614, 2010.

Cory Siler received the B.S. and M.S. degrees in computer science from the University of Kentucky, Lexington, KY, USA, in 2017. They are currently working toward the Ph.D. degree at the University of Kentucky, where they are a student and a Research Assistant.

Their research interests include computational narrative, knowledge representation, and artificial intelligence ethics.

Stephen G. Ware (M'08) received the B.S. degree in both computer science and philosophy from Loyola University, New Orleans, LA, USA, in 2008, the M.S. degree in computer science from North Carolina State University, Raleigh, NC, USA, in 2011, and the Ph.D. degree in computer science from North Carolina State University in 2014.

Currently, he is an Assistant Professor of Computer Science and director of the Narrative Intelligence Lab at the University of Kentucky, Lexington, KY, USA. His research interests are in computer narrative, especially automated planning, knowledge representation, narratology, cognitive science, and artificial intelligence in games.

Dr. Ware is a member of the Association for the Advancement of Artificial Intelligence, the International Game Developers Association, and the Association for Computing Machinery.