Proceedings of the Nineteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2023)

Enhancing Character Depth through Personality Exceptions for Narrative Planners

Elinor Rubin-McGregor, Brent Harrison, Cory Siler

Department of Computer Science University of Kentucky
Davis Marksbury Building, 329 Rose Street
Lexington, KY 40506-0633 USA
erru227@uky.edu, bha286@g.uky.edu, jcsi225@g.uky.edu

Abstract

In the field of narrative planning, implementing character personality is a challenge that's been tackled many different ways. Most of these methods do not incorporate any method for personality to shift when characters are put in situations that would, through stress or satisfaction, naturally cause the character to behave differently than usual. Through use of situationally-triggered *Personality Exceptions*, we can support the generation of a story that prominently features such personality shifts as a narrative tool. This feature is made as generic as possible so that it can be attached onto a wide range of personality models in narrative generators. Through adapting Indexter's indexes of narrative salience towards tracking internal narrative salience in the characters' memories, we can accurately pinpoint triggers which are used to activate these personality exceptions in thematically relevant situations.

Introduction

Several approaches have been made to enable narrative generators to consider the personalities of various characters in generating a story, such as Bahamón, Barot, and Young's goal-based personality model (Bahamón, Barot, and Young 2015), and Shirvani and Ware's choice-based personality model (Shirvani and Ware 2019). However, one limitation is that many of these narrative generators treat personality as static and unchanging, rather than supporting complex personality dynamics.

To support a flexible personality model in response to external stimuli, we have created **Personality Exceptions**, a combined index holding a narrative-internal situational **trigger**, and an associated personality modification **effect** to be applied when a given character is exposed to the aforementioned trigger. Our goal is to enable character depth in narrative generators by creating internal situation-based exceptions to character personality that activates based on an event in the present that is related to some highly salient past event that the character has experienced. Our approach takes Indexter's concept of narrative salience, which was originally proposed to measure memorability of events to a reader, and uses it instead to measure memorability of events to a character (Cardona-Rivera et al. 2012). We can then simulate the influence of powerful pre-existing character memories from the character's backstory by assigning a shift to the character's personality that only applies when they "recall" these memories, as indicated by encountering an event that scores highly along one of Indexter's indices. This allows for a slew of complex behaviors that are seen often in media: A cruel and malicious antagonist has one friend they are kind towards. Inversely, an all-loving protagonist has one opponent they despise due to personal vendettas. As in the case of *The Glass Menagerie*, a terminally shy girl becomes self-confident only in the presence of her figurine collection. We can assume that Laura has countless memories of many hours of calmly enjoying her figurines, though the audience needs not to have this detail pointed out to understand that she loves her collection.

For our purposes, we first outlined a set of narratives wherein understandable situational triggers causing shifts in behavior are critical to the story's conclusion. These served as our baseline story, showing what narratives are generated when personality is not modeled. We then added personality attributes to characters with strict prohibitions on behavior not in line with a character's personality model. In doing so, we expected the generator to struggle or even outright fail to produce a narrative that fit all expectations, as the domain relied upon one character behaving in different ways at different points in the story. Finally we created further modified domains with specific personality exceptions featured, but also still using the strict personality limitations mentioned previously. To ensure these triggers only activated in the proper situations, we adapted Indexter's narrative salience indexes to simulate internal character memory. This last group we expected to succeed, as the personality exceptions allowed characters to behave differently if put in the right situations. In this paper, we show that personality exceptions will support narratives with greater character depth.

Related Work

The base idea for this contribution began with the idea to take Indexter (Cardona-Rivera et al. 2012)—a model of narrative salience in the audience's memory—and apply it to create character depth in generated narrative. Using indexer we've come up with ways that the model can identify present events that are related to highly salient past events

from the perspective of the internal agents controlled by the model. To this end, we focused on various implementations of both personality and narrative salience. The Glaive narrative planner (Ware and Young 2014) produces stories that meet the author's intended narrative goals while honoring that characters would not act against their individual personal goals, in order to create a narrative generator that used believable characters with clear motives understandable to the audience. An extended model where characters have and act on their own (possibly inaccurate) beliefs about the world (Shirvani, Ware, and Farrell 2017; Shirvani, Farrell, and Ware 2018) was implemented in another narrative planner, Sabre (Ware and Siler 2021).

Several key features from Glaive play a heavy role in our work. To begin with, we use the concept of a *Proposition* as a statement about the internal world of the narrative being generated that can be changed through the events of the story. Our planner works by generating a tree of potential stories, each constructed as a sequence of *actions* occurring in a specific order- each action taking one timestep of time. Each *action* taken must be defined by the author in the domain as a potential *action*, and has three components to it:

- A set of *preconditions*- propositions that must be true at the beginning of the *timestep* at which the action occurs in order for the action to be permitted
- A set of *effects*- statements about the world that become true after the action is performed
- A possibly-empty set of *consenting characters*- characters who must all "give permission" for the action to occur

The concept of a *causal link* is a connection between any two events in the narrative, i and j, that only exists if the event which occurred first is responsible for enabling the later action to occur. That is, if $i \rightarrow j$ and j has a precondition that was most recently made to be true as an effect of i. The set of all preceding events in all causal links that can be tracked backwards from any given event i are called the set of *causal ancestors* of that event i.

In addition to these, Glaive distinguished between the concepts of *author goals* and *character goals*, collections of propositions that the author or character "want" to be true. The planner will attempt to generate a story that meets all authorial goals, while only permitting characters to be *consenting characters* for actions that help make their character goals true (Ware and Young 2014). Characters are allowed make multi-step plans known as *character plans* at each timestep to progress their goals (Shirvani, Farrell, and Ware 2018).

The concept of implementing personality into story generation planners has been approached in many ways. Bahamón and Young (Bahamón and Young 2013) proposed an algorithm using the Big Five personality structure for simulating character personality. A different personality structure, OCEAN, was implemented on a modified format of Glaive (Shirvani and Ware 2019). Attempts at long term character growth have been made through games like Dwarf Fortress and the interactive drama *Versu* (Evans and Short 2014), but these cases lack the depth of exceptions because changes in a character's personality are applied universally the entire character and carry through all situations for overall personality change. Our work focuses on a specific feature of a character's depth, rather than growth, in that the personality shifts caused by exceptions are explicitly temporary, which allows for subtlety in the character's personality without changing the entire character. Other models, such as McCoy's narrative system (McCoy 2014) and the work in Kreminski's *Why Are We Like This* (Kreminski et al. 2020) do modify behavior based on models of character memory of past events, however these approaches only change behavior during direct interaction with or actions related directly to specific other characters. These results are similar to our "protagonist" index exceptions, but any non-social influences upon the character's behavior in these systems are not supported. Furthermore, these systems lacks support for indirect influences on character behavior as both of our "causal" and "intention" indexes do.

The computational model for event-indexing known as Indexter was introduced to analyze audience memory salience of previously seen events in a narrative (Cardona-Rivera et al. 2012). Given some current timestep in the narrative, Indexter maps out how likely the audience is to recall specific previous scenes of the narrative based on similar or shared features between the current scene and the specific previous scene. The idea to use Indexter in a narrative planner was first introduced by Farrell and Ware (Farrell and Ware 2016). Farrell and Ware proposed using the Indexter model for the purpose of predicting player choices in interactive narratives, hypothesizing that players were more likely to take certain actions depending on which previous details they remembered from earlier in the story. In this way, an interactive narrative could subtly guide players towards choosing certain options to lead to more favorable outcomes. Our proposed work essentially takes this concept and transforms it into an internal prediction, modeling theoretical memory salience for characters based on hypothetical events in the "backstory"- that is, taking place before the very first event in the story generated. In short, our model uses Indexter's indexes to model memories salient to the characters rather than the audience or players.

Methods

Recall that our primary contribution is a method for enabling fluidity in a character's actions with respect to their personality via *personality exepctions*. When constructing narrative plans, we check to see if each character's behavior aligns with their established personality. By using personality exceptions, we enable behaviors that may appear contradictory to a character's personality to be acceptable if it is deemed *applicable* by our personality exception measures. Our work towards creating personality exceptions primarily involved finding machine-calculable internal scenario-based triggers to simulate character memory salience. At this time, the internal highly salient memories that cause Personality Exceptions are assumed to originate in "events" that occurred prior to the story's setting. That is, all Personality Exceptions are currently defined in the domain of the story, with the narrative explanation being that the Personality Exceptions were

caused by events in the related agent's backstory.

To determine if an action is salient to the character, we use the Indexter model. In the original Indexter model (Cardona-Rivera et al. 2012), five indexes are evaluated to estimate memory salience. These were defined as:

- Time: time of the current timestep using the IPOCL (Riedl and Young 2010) model of time frames
- Space: location of the current timestep
- Protagonist: presence of the narrative's protagonist in the current timestep
- Causal: which causal ancestors can be traced back from the current timestep
- Intention: which character goals are motivating actions that occur in the current timestep (Cardona-Rivera et al. 2012)

For our purposes, we have adapted these to fit internal character memory salience as such on a specific focus character:

- Time: time label of the current timestep
- Space: location of the current timestep
- Protagonist: the presence or investment of a specific character other than the focus character in the event
- Proposition: if a given proposition is made true either due to the event at the current timestep or for the sake of the event at the current timestep
- Action: which specific types of actions are occurring in the current timestep
- Intention: the specific goals of the focus character being progressed in the current timestep

The causal index was broken into two different indexes, one for specific trigger actions (as defined in the domains) being taken, and one for specific trigger propositions being found as causal ancestors. These were implemented for flexibility and application in planners where causality cannot be easily traced. We've found that the two results provide unique outcomes and could both be applicable to different cases, and thus have chosen to describe both.

Every Personality Exception must have an $applicable$ function which can gather and report whether the exception should apply for the character to whom it belongs, at a specific timestep given—that is, whether the character's behavior at the given timestep should reflect their normal personality or their exception-caused personality. The details of how the $applicable$ function operates will, by necessity, be unique to each of our six index exceptions. For this function to work, there are certain attributes which every Personality Exception must have:

- c: A reference to the character who the Personality Exception belongs to
- $modified$: A new personality value, personality trait value, or a personality modifier
- $trigger$: the specific type and value of the $trigger$ will be unique depending on which of our six exceptions is used

Algorithm 1: Applicable Function on Time index

Input: time label $trigger$, current timestep t
Output: True/False boolean reflecting exception applicability at input timestep t

```
if time(t) == trigger then
    return True
end if
return False
```

Every single $applicable$ function begins by directly checking applicability through testing if the $trigger$ in question is present to the character with the Personality Exception at the given time. It should be noted that these algorithms all assume that these checks on personality, and therefore Personality Exceptions, only occur for actions which the given character consents to.

Time

The time frame was adjusted to feature some authorial-defined attribute of time that could occur or reoccur in a cycle or at random. For example, the attribute of "morning" could occur as part of a cycle followed in order by the time attributes of "afternoon", "evening", and "graveyard hours". This could be applicable to a story wherein a character is kind but not a morning person, and will act less outgoing and creative during the "morning" times. It should be noted that time is assumed to be universal at a given event. The time value can only be changed with subsequent events, and applies to all locations and characters universally across any timestep where it is in place.

The $trigger$ for a time-index exception will need to be a generic time-attribute, for example "12pm" or "4pm". It should also be noted that this algorithm requires the planner to be able to provide the time attribute's value from a given timestep collected, and there must be a way to accurately compare the generic attribute of the trigger with the current attribute in the planner's timestep.

The algorithm for applicability is rather simple—it only requires performing the direct check to see if the current time is equivalent to the $trigger$ time attribute. The reason for this simplicity, when later functions will check longer-scale character plans, is that with small-scale domains time attributes can be highly limited to perhaps two different time attributes, say "day" and "night." This same issue crops up with locations, as we see in the Space index below.

Space

The implementation of our Space-index exceptions is similar to our Time-index exceptions, as the $trigger$ for a space-index exception will be a generic location-attribute. Unlike Time-index exceptions, location is not assumed to be universal at a given timestep. Much like Time, it's assumed that the author will define a list of Spaces in the domain as well as any rules for transfer between locations. However unlike Time, Space is a trait applicable to specific characters and objects individually. Any object or character which exists in-universe will mark one of the existing locations as the

Algorithm 2: Applicable Function on Space index

Input: location $trigger$, current timestep t, character the Personality Exception belongs to c
Output: True/False boolean reflecting exception applicability at input timestep t

if $location(c, t)$ == $trigger$ **then**
  return $True$
end if
for $eff \forall effects(t)$ **do**
  if $trigger \in eff$ **then**
    return $True$
  end if
end for
return $False$

Algorithm 3: Applicable Function on Protagonist index

Input: character $trigger$, current timestep t, character the Personality Exception belongs to c
Output: True/False boolean reflecting exception applicability at input timestep t

for $eff \forall effects(t)$ **do**
  if $trigger \in eff$ **then**
    return $True$
  end if
end for
if $utility(trigger, t) \neq utility(trigger, t-1)$ **then**
  return $True$
end if
for $step \forall plan(c, t)$ **do**
  for $eff \forall effects(step)$ **do**
    if $trigger \in eff$ **or** $utility(trigger, t) \neq utility(trigger, t-1)$ **then**
      return $True$
    end if
  end for
end for
return $False$

"current" location for each given timestep. So character A can start the narrative in Italy while character B starts the narrative in France. If the first event is for character A to fly to France, then at the second timestep character A and character B will both be in France.

The Space-index exception checks direct applicability with the location of the agent to whom the exception belongs. It then checks to see if the action taken in the current timestep will result in changes that apply to the $trigger$ location, for example if the current action has the character traveling to the $trigger$ location.

To decide if an exception applies during a certain timestep, for this algorithm, assume that $effects(t)$ is the list of all effects produced by the action that occurs at timestep t. Note that use of $i \in j$ (with i and j as any two theoretical variables) will not necessarily be used to indicate that i exists directly in j, but rather than some sub-piece of j contains i inside of it. In the case of Time exceptions, we will see this to mean that some proposition effect j features time i in it. In short, we are using $\in$ to indicate both if i exists in j, and if i can be found in a sub-component of j. Assume that $location(c, t)$ will provide the space where character c is located at timestep t.

Protagonist

The base idea behind the protagonist index exceptions is that there is one person who the acting character treats differently than they do everybody else. Perhaps the cold-hearted villain has a younger sibling they adore, perhaps the pacifistic all-loving hero can't stand the villain who slaughtered their village, perhaps the brilliant genius will lose their head when their lover walks into the room and become unable to think of anything else. These kinds of relationship dynamics have been in stories for countless years, and they add a level of flavor to narratives that's hard to find anywhere else.

The $trigger$ for a protagonist-index exception will be a pointer to another character in the story. A character theoretically could, but most likely should not, have themselves as their trigger character, as in most domains this would likely cause the trigger to apply in almost every case. The direct applicability check works by checking two things. One is if the $trigger$ character is included in anywhere in the set of $effects$ for the action, to see if the action has a direct impact on the trigger character. In addition, if the current timestep's action changes the progress towards the personal character goal of the $trigger$ character, then the exception applies as well. We determine if this progress has changed by evaluating the accuracy of the $trigger$ character's goals at the current timestep, as well as at the timestep of the event that occurred immediately prior to the current timestep in the planned story. If the evaluated accuracy has not changed, then we can assume that the $trigger$ character's goals have not been progressed or hindered. For character triggers, we also consider the plans of the acting agents as well as the impacts of the current timestep.

For this algorithm, all prior syntax applies. Assume that $utility(a, t)$ refers to a numeric value reflecting how many goals of character a are met at timestep t. Assume $plan(a, t)$ provides a list of theoretical timesteps of character a's character plan held at timestep t, each with the same attributes and functionalities as t. Likewise, assume that $utility(a, t-1)$ will reflect how many goals of character a are met at the event preceding the event that occurs at timestep t.

Proposition

The causal ancestor exception is made for a specific state about the world that personally affects the character to whom the exception belongs. Although this is not a requirement, the concept behind this exception is that the exception-holding character either wants or does not want the proposition to be true—for example, stories featuring an antagonist who normally abhors violence but is willing to commit murder if it progresses their plot for world peace. A strategic-minded officer might become emotional and short-sighted when their own hometown is endangered. A lazy adventurer might avoid taking the easiest route to treasure if it requires

Algorithm 4: Applicable Function on Causal Ancestor index

Input: proposition $trigger$, current timestep t, character the Personality Exception belongs to c

Output: True/False boolean reflecting exception applicability at input timestep t

if $prop(t, trigger)$ **then**
  return $True$
end if
for $eff \forall effects(t)$ **do**
  if $trigger \in eff$ **and** $c \in cons(t)$ **then**
    return $True$
  end if
end for
for $step \forall ancestors(t)$ **do**
  if $prop(step, trigger)$ **and** $c \in cons(step)$ **then**
    return $True$
  end if
  for $eff \forall effects(step)$ **do**
    if $trigger \in eff$ **and** $c \in cons(step)$ **then**
      return $True$
    end if
  end for
end for
for $step \forall plan(c, t)$ **do**
  if $CausalAncestor(step) == True$ **then**
    return $True$
  end if
end for
return $False$

disturbing their own mother's grave. Almost every person alive has opinions on how the world should or should not be. The proposition index is intended to allow narrative generators to portray characters who behave differently when some deeply-held belief of theirs is threatened.

The $trigger$ for a proposition-index exception will be a proposition, of course. For this to work, the planner must be able to report if, at a given timestep, a provided proposition is true or not. We assume that negated propositions are allowed, as our exception will be considered applicable if the $trigger$ is true in a given timestep, or made true as a result of the event at that timestep. In addition, the Proposition index also collects causal ancestors of the current timestep's event, and returns true if any causal ancestors of the event at the current timestep require this proposition. Finally, if any planned event which the agent consents to contains the proposition as an effect or causal ancestor, the exception is deemed applicable.

Assume $prop(t, p)$ will return true if the proposition p is true at timestep t and false otherwise. Assume $ancestors(t)$ will return a list of timesteps at which all causal ancestors of the event at time t. Likewise, assume $cons(t)$ is the list of all characters which consent to the action that occurs at timestep t.

Algorithm 5: Applicable Function on Causal Action index

Input: potential action $trigger$, current timestep t, character the Personality Exception belongs to c

Output: True/False boolean reflecting exception applicability at input timestep t

if $act(t)$ /= $trigger$ **then**
  return $True$
end if
for $step \forall ancestors(t)$ **do**
  if $act(step)$ /= $trigger$ **then**
    return $True$
  end if
end for
for $step \forall plan(c, t)$ **do**
  if $act(step)$ /= $trigger$ **and** $c \in cons(step)$ **then**
    return $True$
  end if
end for
return $False$

Action

The action index is more for quirky characters, but it's still a valid one. For planners without a way to collect causal ancestors, it may be the only way to implement a causality-focused Personality Exception. This one triggers if the character is going to take a certain kind of action. Characters with neuroses would behave differently if permitted to engage in their passion. Batman is willing to use violence of all kinds against criminals, but he will not fire a gun.

Notably, while we care if a planned event features the exception-holding character taking part in an action of the given generic action $trigger$, we don't care if the causal ancestors contain a $trigger$ event that this character consented to—only that it occurred. This is an optional feature that can be worked around, but it was included so as to support stories that feature situations where a character is either emboldened or discomforted by the knowledge of what was done to enable their current actions. For example, a dog-loving hunter might struggle to use a bear trap if their own childhood puppy lost a leg to a bear trap, but they also might hesitate to kill a bear found in one such trap as the bear might remind them of finding their dog in such a state. In the former case we'd have the exception applied due to the $trigger$ matching the current event, but in the latter case it would be a precondition to killing the bear—the bear being injured by the trap—that would enable the exception to activate.

For this algorithm, the format $a/ = b$ will be used to indicate the concept of "a is an example of b", where a and b are both actionszz. Likewise, assume $act(t)$ will return the action occuring at timestep t.

Intention

The narrative concept behind the intention index exception is a character whose motivation to reach a certain goal is so strong that they change their behavior when acting for that purpose. In order for the exception to work properly, the planner should check the character's existing goals at

Algorithm 6: Applicable Function on Intention index

Input: goal $trigger$, current timestep t, character the Personality Exception belongs to c

Output: True/False boolean reflecting exception applicability at input timestep t

if $eval(trigger, t) > eval(trigger, t-1)$ **then**
  return $True$
end if
for $step \forall\, plan(c, t)$ **do**
  if $eval(trigger, step) > eval(trigger, step-1)$ **then**
    if $ancestors(step)$ contains t **then**
      return $True$
    end if
  end if
end for
return $False$

creation and only allow intention exceptions to be made for goals that the character holds. In that way, it's ensured that the generator will be motivated to have the character act towards this goal before checking for personality.

It should be noted that there are conceptually some similarities between Intention index exception and proposition index exceptions—both involve a character having something they want to happen or not happen. We chose to break these into different categories in order to support narrative generators where character goals were represented in some non-proposition based format. Shirvani's emotion-based personality system, for example, utilizes emotional states instead of direct proposition goals (Shirvani 2021).

The $trigger$ for a intention-index exception should be the same class as a character's goals—ideally an equation that can be evaluated at any given timestep for a value representing how close that goal is to being met. The direct application check merely sees if the trigger goal has been progressed by the action- as it is assumed characters would not consent to actions which hinder any of their goals, or if they do then their actions cannot be considered to be motivated by those specific hindered goals. In planners where character plans are available, any planned step that helps progress the $trigger$ goal should be checked to see if it takes the current step as a causal ancestor. If so, then it can be assumed that the current action is at least partially motivated by the $trigger$ goal and thus the exception may be applied.

Assume all prior syntax still holds. For the algorithm below, assume that $eval(g, t)$ will return the value for goal g evaluated at timestep t. Likewise, assume that $eval(g, t-1)$ will return the value for the goal g as evaluated at the timestep that equates to the event that occurred immediately before the event occurring at timestep t.

Experiments

As proof of concept, we sought to provide examples of narratives where character depth is not supported by static personality models, but is supported when Personality Exceptions are added in. To prove that static personality models do not support such narratives, we set hard limitations on our narrative generator so that any behavior considered "out of character" would be completely forbidden. We then outlined a set of narrative planning domains wherein understandable situational triggers causing shifts in behavior are critical to the story's conclusion. Our expectation was that when running with static personality the generator would struggle or even outright fail to produce a narrative that fit all expectations, as the domain relied upon one character behaving in different ways at different points in the story. We used domains expected to produce shorter stories in order to isolate Personality Exceptions and their performance in the results. We then added related Personality Exceptions to the narratives to show that the inclusion of Personality Exceptions allows the narrative generator to find a satisfactory story.

For our experiments we applied a modified variant of Shirvani's OCEAN based personality model to the Sabre belief-intentionality intersecting state-space narrative planner (Ware and Siler 2021). Our set of experiments were focused on proving that the exceptions could work to improve the range and quality of personality-implementing planners. To that end, we used a planner where any plan that portrayed a character behaving against their personality was outright rejected and removed. Our personality model used five numeric values for each of the OCEAN personality traits, ranging from zero to one. We set the default personality values for each character to 0.5 and then set the threshold for out-of-character behavior plan rejection to 0.5, so that any personality attributes not being tested would never cause plan rejection. We ran a unique collection of domain sets that were all meant to represent the same basic "story" with different features included or excluded to show the impact Personality Exceptions had on the results. Collectively, we call this the *hollow* domain set.

The *hollow* domain set picks one OCEAN trait to make relevant, specifically Agreeableness, for the sake of simplicity. Agreeableness can be summarized as how compassionate and considerate of others a character behaves. Actions which feature multiple consenting parties and progress the goals of other characters have high Agreeableness scores, while actions that hinder other characters goals and have few consenting parties have low Agreeableness scores.

The *hollow* story has three characters: Shadow, Sara, and the Sprites. A proposition known as "suffer" exists for every character, such that at any given timestep each character can have "suffer" true or false for them. Shadow wants the Sprites to suffer, while Sara and the Sprites want the opposite. Shadow also wants Sara to be alive, and Sara is the only mortal character of the three, meaning she will die if she suffers. Our authorial goals are for Shadow to become mortal, and the Sprites to not "suffer". It should be noted that this domain set fixates on what would ideally be the climax of a longer work, and a longer narrative would portray each character's base personalities outside the influence of Personality Exceptions, so that the emotional impact on the audience would be more meaningful during parts of the story where the Personality Exception(s) are active. We used shorter domains in order to isolate the impact of Personality Exceptions, but also to give room for future work to pro-

duce Personality Exceptions during narrative generation by allowing an event generated by the planner to be identified as highly salient to a character and used to produce a Personality Exception.

It should be noted that in order to keep our experiments as simple as they could be, we incorporated the elements of multiple locations and shifting times only in the domains that were testing these elements. But as our goal was to keep our domain set as similar as possible, any deviations in domains warrented a unique test to ensure that these changes were not causing unexpected deviations. To that end, we created three near-identical domains with no personality, only differing in the inclusion and exclusion of spatial and temporal implementation. For example, the test hollow1 incorporates time, while hollow2 and hollow3 do not. Likewise in implementing personality, we extended these three base tests in the same ways. So hollow2.1 has multiple locations and personality (but no personality exceptions), while hollow1.1 and hollow3.1 both have personality neither incorporates multiple locations.

Hollow Domain Experiments

No Personality When no personality is included (hollow3 in Table 1), the story reaches that conclusion in a slightly disturbing way:

1. Shadow torments the Sprites to cause them pain. It should be noted Shadow is capable of tormenting an immortal being only because he is also immortal.
2. The Sprites torment Sara, causing her pain that kills her.
3. Shadow and the Sprites work together to revive Sara, which requires Shadow to sacrifice his immortality and become mortal.
4. Sara sacrifices herself to free the Sprites from their pain. This sacrifice causes her great pain, and because she is mortal, this kills her.

Two additional domains exist- one implementing time in the form of a four-time cycle that loops from "Morning" to "Afternoon" to "Evening" to "Graveyard" and in a cycle after every single event takes place (hollow1 in Table 1). This domain had the same story produced as the original. Another base domain implements two locations, a "Lake" and a "Hollow." We added a handful of extra conditions, such as the "revive" action and "torment" action requiring all relevant parties to be at the same location, and a new action to enable travel. This domain is summarized as hollow3 in Table 1. This domain ran similarly, with one extra step added for the Sprites to take Shadow to the Hollow before tormenting Sara.

Static Personality We created one modified variant of each of our three base domains (hollow1.1, hollow2.1, and hollow 3.1 in Table 1) to implement personality for one character. Specifically, we gave Shadow an Agreeableness score of 0.0, and the Sprites an Agreeableness score of 1.0. The result of this was that with our planner prohibiting any out-of-character behavior, the planner could find no working story with any of our three domains. Notably, even when we removed the modified Agreeableness for the Sprites and only gave Shadow a unique personality score, the domains still failed to find a solution. This is because Shadow's personality bars him from working with other characters and from progressing the goals of other characters. The revive action would require working with the Sprites, and would help achieve Sara's own goal of being alive. Thus with strict personality in place, we can see how a personality-implementing planner would not support stories that require character depth to this degree. Shadow can be a cruel character, but doing so hinders any plot-action that requires him to be kind, even if there's a valid reason in the story for this behavior.

Domain	P	EI	Time	Space	SF
hollow1	No	-	Yes	No	Yes
hollow2	No	-	No	Yes	Yes
hollow3	No	-	No	No	Yes
hollow1.1	Yes	-	Yes	No	No
hollow2.1	Yes	-	No	Yes	No
hollow3.1	Yes	-	No	No	No
hollow1.1.1	Yes	Time	Yes	No	Yes
hollow2.1.1	Yes	Space	No	Yes	Yes
hollow3.1.1	Yes	Protag	No	No	Yes
hollow3.1.2	Yes	Prop	No	No	Yes
hollow3.1.3	Yes	Action	No	No	Yes
hollow3.1.4	Yes	Action	No	No	Yes
hollow3.1.5	Yes	Intent	No	No	Yes

Table 1: Domains by included factors. Key: P: Personality, EI: Exception Index, SF: Solution Found

Personality Exceptions With that, we created six further developed domains, each of which implemented a different Personality Exception, and each of which successfully found a story that satisfies the author's goals. Based on our assumption that it's Shadow's low Agreeableness score that prevents an in-character story that meets authorial goals from being reached, we added exactly one Personality Exception to each case. While the triggers differed, every exception belonged to Shadow, and when triggered the exceptions would change his Agreeableness score to 0.5. These six domains all produced working stories, and since the only change was the inclusion of Shadow's character depth, we can confirm our earlier assumption that the personality-only narrative is hindered due to Shadow being unable to revive characters.

Time Index Exception When we added in a Personality Exception for Shadow along the Time Index (hollow1.1.1 in Table 1), we found a valid story that did not feature cruel behavior from the kind-hearted Sprites, and allowed the cruel Shadow to do something for others. Our exception chose the *Graveyard* time as the trigger, the narrative idea being that Shadow becomes contemplative and compassionate during the Graveyard hours of the night. This produced a very dif-

ferent story:

1. ***Morning:*** Shadow torments the Sprites
2. ***Afternoon:*** Sara sacrifices herself to free the Sprites from their pain, ultimately causing her death
3. ***Evening:*** Shadow torments the Sprites again, causing them more pain
4. ***Graveyard:*** Shadow and the Sprites both revive Sara, rendering Shadow mortal
5. ***Morning:*** Sara again sacrifices herself for the Sprites

This tale's differences show how personality can be useful in creating unique and interesting stories. The original story implied that the Sprites are highly amoral and calculating, tormenting Sara to manipulate Shadow into giving up his immortality, while at the same time planning for Shadow's goals to fail after Sara sacrifices herself for them. Using personality gets a story where the Sprites are much more compassionate, which may be closer to the author's goals.

Between the personality-lacking and exception-handling domains the narratives are quite different, and while we might've been able to collect the second story through adding on extra goals and loopholes, personality implementation provides a shortcut that's much easier and will encompass all possibilities. For example, we might've gotten the second story without personality by giving the Sprites goals to want Sara alive. But if we were looking at this small story as one piece of a larger story, that might not be so easy to generate- there would be more characters to include, and to portray the Sprite's compassion their goals would need to include desires for the survival of every single character. Simply giving them a high compassion score covers that easily as well as other aspects of compassion.

Space Index Exception For the Space Index Exception (hollow 2.1.1 in Table 1), we chose to use the location of "Hollow" as the trigger for Shadow's agreeableness. This way Shadow would still display cruel tendencies at the beginning of the story when he and the Sprites are both at the Lake. Since a sacrifice could only happen at the Hollow we predicted that Shadow and the Sprites would need to travel to the Hollow for the story to work. When the exception set Agreeableness to 0.5 as normal, the story diverged very little from our predictions. The first event changed to have Shadow take the Sprites from the Lake to the Hollow, and all subsequent events followed the same path as the story created by the Time Index Exception domain.

Protagonist Index Exception The domain featuring an exception along the Protagonist Index (hollow 3.1.1 in Table 1) worked by giving Shadow a Personality Exception with Sara as the target. When triggered, the exception changed Shadow's Agreeableness index to 0.5, thus showing that Shadow is willing to cooperate with others (though he won't like it) if Sara is involved. We got the same story as generated with the Time Index Exception, but this time the narrative implications are different.

In the Time-index exception and Space-index exception domains, Shadow is normally cruel unless certain conditions are met. Given this is a story with immortals, these conditions could well be supernatural in nature—perhaps the graveyard hours have mystical properties that soothe Shadow's soul, or perhaps the Hollow is a sacred place whose holy energies influence Shadow. In the story generated by protagonist index exception, the most straightforward explanation is a tale driven by character flaws. Shadow chooses to abuse the Sprites even after Sara's sacrifice, and while Shadow is willing to sacrifice his immortality for Sara, his own cruelty renders the point moot when she simply gives her life a second time.

Proposition Index Exception The domain displaying the proposition index exception (hollow3.1.2 in Table 1) utilized a trigger that was deliberately picked to be as close to the original concept as possible, and thus was simply the condition that Sara is alive. The result matched the Time Index Exception exactly, although we can argue that the narrative reflects a unique story. Rather than Shadow being kind towards Sara, it could be that Shadow needs Sara alive for some unknown reason and is simply pragmatic enough towards this goal that he's willing to do whatever it takes to reach it.

Action Index Exception For the Action index, we found two different tests that produced the same story as the Time index exception domain, but used different triggers of revive (hollow3.1.3 in Table 1) and sacrifice (hollow3.1.4 in Table 1) actions. Narratively these exceptions could be explained by expanding Shadow's backstory to give him a fascination with these two highly Agreeable actions.

While revival is an action that Shadow partakes in himself, the sacrifice action is one that he has no role in. What happened in this case is that a revival includes the target's dead status as a precondition, which makes the event that led to that status a causal ancestor. The fact that the domain works both ways proves that the causal ancestor check is indeed useful, and also illustrates why it is necessary to add that feature.

Intention Index Exception The Intention index exception domain (hollow3.1.5 in Table 1) used Shadow's goal to keep Sara alive as its trigger, and produced the same story as the Time index exception domain. Though the results are largely the same, it's still noteworthy that the concept works. In a longer narrative, the use of Intention index rather than Protagonist index could enable a story wherein Shadow has some selfish motivation to want Sara alive, but does not care for her well being as a person.

Future Work

Several aspects of the experiment that can be improved upon and explored in greater depth. One prominent area that we lacked the time to delve into is fully applying the exceptions to more than one personality-implementing narrative planner, to prove that the algorithms can be generalized. While the label-based time exception introduced has its own benefits, using an exception that implements the Indexter's original IPOCL frame model could also produce interesting results. Another expansion could be to include multiple-index personality exceptions, to allow multiple avenues by which

a single character could be influenced by past history. This would allow the exception in question to have more influence on the story. Research on emotions has indicated that utilizing multiple memory triggers may be a more accurate reflection of real-life applications (Moors et al. 2013), and this feature could also increase the audience's perception of the specific aspect of character depth the exception represents in longer narratives that may not touch on the exception frequently.

In terms of long-term application, there are some unique implications of this project. A logical next step would be enabling the planner to generate Personality Exceptions through including significant events that would naturally cause some lasting psychological effects on the characters. That is to say, finding a way for the planner to generate causes for Personality Exceptions would be an ideal continuation. In the example of our experimented domain, we could remove the pre-existing exception and instead have the planner add on an additional first step where Sara protects or helps Shadow in some regard. This event would give him motive to show her kindness in return, and allow the planner to assign a Personality Exception to him after that event. To do so one would most likely need some way of measuring a score to represent intense emotional attachment- likely some measurement of changes in how well the character's goals are met and relevance of elements in the preconditions of the event that causes the shift. This is a future project, and details won't be explored here.

If we can make Personality Exceptions both scalable and generatable by a planner, this opens up narrative planning for stories where character development is the primary focus. If a character gains enough Personality Exceptions with the same resulting personalities, or if they gain an exception with a trigger that is usually in place, the character after gaining the exception(s) will eventually effectively have a different personality—but since the audience got to watch the character develop this way, the change feels natural.

Coming-of-age stories, redemption stories, stories about a villain protagonist's fall to darkness—these genres are many and well loved by audiences. While they could be somewhat emulated by planners without personality modeling using strict goals, personality ensures consistency and allows the planner to come up with unique stories outside of the author's predictions that still fit the character's personality and may even work better than what the author originally intended.

References

Bahamón, J.; Barot, C.; and Young, R. 2015. A Goal-Based Model of Personality for Planning-Based Narrative Generation. In *AAAI Conference on Artificial Intelligence*, volume 29, 4142–4143.

Bahamón, J. C.; and Young, R. M. 2013. CB-POCL: A Choice-Based Algorithm for Character Personality in Planning-based Narrative Generation. In *Workshop on Computational Models of Narrative*.

Cardona-Rivera, R. E.; Cassell, B. A.; Ware, S. G.; and Young, R. M. 2012. Indexter: a computational model of the Event-Indexing Situation Model for characterizing narratives. In *Workshop on Computational Models of Narrative*, 34–43.

Evans, R.; and Short, E. 2014. Versu—A Simulationist Storytelling System. *IEEE Transactions on Computational Intelligence and AI in Games*, 6(2): 113–130.

Farrell, R.; and Ware, S. G. 2016. Predicting user choices in interactive narratives using Indexter's pairwise event salience hypothesis. In *International Conference on Interactive Digital Storytelling*, 147–155.

Kreminski, M.; Dickinson, M.; Mateas, M.; and Wardrip-Fruin, N. 2020. Why Are We Like This?: Exploring Writing Mechanics for an AI-Augmented Storytelling Game. In *International Conference on the Foundations of Digital Games (FDG '20)*.

McCoy, e. a., Joshua. 2014. Social story worlds with Comme il Faut. *IEEE Transactions on Computational intelligence and AI in Games 6.2*, 6(2): 97–112.

Moors, A.; Ellsworth, P.; Scherer, K.; and Frijda, N. 2013. Appraisal Theories of Emotion: State of the Art and Future Development. *Emotion Review*, 5: 119–124.

Riedl, M. O.; and Young, R. M. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research*, 39: 217–268.

Shirvani, A. 2021. *Personality and Emotion for Virtual Characters in Strong-Story Narrative Planning*. Ph.D. thesis, University of Kentucky.

Shirvani, A.; Farrell, R.; and Ware, S. G. 2018. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 222–228.

Shirvani, A.; and Ware, S. G. 2019. A plan-based personality model for story characters. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 188–194.

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

Ware, S. G.; and Siler, C. 2021. Sabre: A narrative planner supporting intention and deep theory of mind. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 17, 99–106.

Ware, S. G.; and Young, R. M. 2014. Glaive: a state-space narrative planner supporting intentionality and conflict. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 80–86.