MULTISTYLE: Characterizing Multiplayer Cooperative Gameplay by Incorporating Distinct Player Playstyles in a Multi-Agent Planner

Eric W. Lang[1], **R. Michael Young**[1,2]
[1]School of Computing, University of Utah, 201 Presidents' Circle, Salt Lake City, UT 84112
[2]Entertainment Arts and Engineering Program, University of Utah, 201 Presidents' Circle, Salt Lake City, UT 84112
ewlang@cs.utah.edu, young@eae.utah.edu

Abstract

This paper presents MULTISTYLE, a multi-agent centralized heuristic search planner that incorporates distinct agent playstyles to generate solution plans where characters express individual preferences while cooperating to reach a goal. We include algorithmic details, an example domain, and multiple different solution plans generated with unique agent playstyle sets. We discuss our intent to incorporate this planner in a tool for game level designers to help them anticipate and understand how teams of players with distinct playstyles may play through their levels. Ultimately, MULTISTYLE generates solution plans with a novel and increased expressive range by attempting to satisfy sets of action and proposition preferences for each agent.

Introduction

When listening to a group of players playing a video game, it's common to hear them express distinct desires: "I want to build a base!", "I want to kill monsters!", "I want to solve puzzles!", or "I want to go to new places!". In any given video game, players may uniquely prefer doing some activities over others, as each player simply has fun doing different things. The notion of distinct player types is well-known: Bartle's player types (Bartle 1996) and the types defined in Robin's Laws of Good Game Mastering (Laws 2002) are commonly referenced, and research has been published that classifies players as well (Bateman, Lowenhaupt, and Nacke 2011; Kallio, Mäyrä, and Kaipainen 2011; Tseng 2011; Hamari and Tuunanen 2014; Vahlo et al. 2017; Mora et al. 2019). All of these classifications are broadly based on players having different preferences when it comes to their game experience.

We refer to what a specific player prefers to do in a game as a player's *playstyle*. Designers often attempt to satisfy various playstyles when designing single-player levels, producing agency-rich gameplay where players are free to choose to do what they want to do within the game. Ideally, every type of player in the designer's audience can enjoy the game how they want to enjoy it. However, in a multiplayer level, satisfying multiple playstyles simultaneously can be challenging, as the designer needs to consider how groups of players may play through a level when each player in the group has a distinct playstyle. This quickly becomes an unwieldy combinatoric issue: what if a group has one fighter, one builder, and one gatherer? What if a group only has three fighters? What if the group has two explorers and a builder? Quickly enumerating the types of groups and understanding the experience each group will have can be challenging for a human designer even in fairly simple domains.

In this paper, we introduce a playstyle-aware planner named MULTISTYLE, aimed at solving cooperative multi-agent planning problems to indicate what players with different playstyles might do as they work together to complete a game level. Each agent has a playstyle which is composed of action preferences (what the agent wants to do) and proposition preferences (what properties of the world the agent prefers). The planner incorporates this information in its heuristic and produces solutions biased toward each individual agent satisfying its specific playstyle. These solution plans can vary significantly based on the playstyles given to each agent, meaning the planner can express distinct agent behavior. In order for a domain author to inspect multiple possible approaches for agents to reach the goal, they need only modify the playstyle parameters for each agent.

MULTISTYLE is unique in that typical preference-based planners only consider a single preference set that solely holds preference information about the world state (e.g. as defined in PDDL3's preferences (Gerevini and Long 2005)). These preferences are intended to give the user control over what sort of solution the planner will produce. The approach of giving each agent its own preferences with the intent of having agents express their individuality is entirely novel, and by using a method to incorporate both action and proposition preferences instead of solely proposition preferences, resulting solution plans can show idiosyncratic agent behavior.

This paper includes a description of the planner, how the single-agent preference-based heuristic RPGPREF (Lang and Young 2022) was modified to work in a multi-agent context, and an example domain with multiple solution plans illuminating how the planner has the power to generate playstyle-conforming paths to goal. We provide a brief discussion of the domain and associated playthroughs to give a characterization of the increased expressive range provided by the planner. MULTISTYLE can generate multiple different playthroughs from the same planning problem for teams of cooperating agents who each have distinct playstyles. We discuss using that capability for aiding game level designers by providing them with information about the possible ways players with varying playstyles may play through their levels.

Related Work

There are three broad categories of adjacent work for this planner: preference-based planners, multi-agent planners, and planners that can express character traits (broadly referred to here as ”narrative planners”). This planner sits somewhere between the capabilities of all three types of work and each subsection will discuss how this paper is influenced by these fields.

Preference-Based Planners

From a purely conceptual level, preference-based planners are the nearest field to the work presented in this paper, as both preference-based planners and this paper discuss planners which use additions to a typical planning problem to guide the generation of a solution plan toward potentially longer solutions that instead try to fulfill added criteria. However, current preference-based planners are single-agent. Torreño et al. (2017) state ”Preference-based MAP is an unstudied field that can be interpreted as a middle ground between cooperative and self-interested MAP, since it involves a set of rational agents that work together toward a common goal while having their own preferences concerning the properties of the solution plan.” In the years since that paper was written, the field has remained unstudied.

In addition, preference-based planners frequently use PDDL3’s preferences (Gerevini and Long 2005) which only allow for proposition preferences (Baier, Bacchus, and McIlraith 2009; Coles and Coles 2011). This sole focus on proposition preferences is powerful in that the user can guide the solution toward certain world states, but the user has no say in what actions the solution plan should use to reach those world states. One exception to this is RPGPREF (Lang and Young 2022), a heuristic that was developed specifically to incorporate action preferences alongside proposition preferences. Our approach in this paper uses a modified version of RPGPREF to incorporate multiple distinct agent playstyles.

Multi-Agent Planners

Multi-agent planners describe a broad field with many different algorithmic approaches. Recent work focuses on various subfields such as distributed planning, agents using resources, and privacy between agents (Nissim and Brafman 2014; Torreno, Onaindia, and Sapena 2014; Maliah, Shani, and Stern 2017; Gerevini et al. 2019; Ye et al. 2020). Our approach in this planner is centralized, agents have no specific resources or capabilities, and the planner is allowed to be omniscient with no enforced privacy between agents.

This dramatically simplifies the planning process, but this simplification is because of the intended use of our planner. While some multi-agent planners are intended to solve problems where, for example, corporations must work together without exposing their internal operations, the class of problems MULTISTYLE is intended to solve assumes that cooperating players within a video game can freely communicate and be observed without worry of exposing valuable information.

Instead, MULTISTYLE has the novel capability of considering agent preferences in a multi-agent planning process. By making the planner centralized, we remove the ability to consider privacy between agents, but we gain significant capabilities in choosing which agent performs which action. Because our planner is intended to express agent playstyle within its solutions, this ability to consider all agents simultaneously is invaluable as it allows the planner to directly compare between all agents’ playstyles when considering the next step for the plan.

Narrative Planners

In narrative planning, the objective of the planner is to generate a solution plan for a planning problem which can then be used as a story or otherwise incorporated into interactive media such as video games. Significant work has been done in narrative planning to incorporate both authorial goals as well as believable agent behavior.

Authorial goals can be thought of as a requirement for certain states to hold true in the world at certain points (e.g. Darth Vader and Luke must only fight once) which are similar to user preferences for the solution plan. Just like as mentioned in preference-based planning, approaches using this feature differ from our approach in that MULTISTYLE uses a preference set for each agent, not for the entire solution plan. In addition, MULTISTYLE attempts to satisfy playstyles but will still find a solution plan even if it can not wholly satisfy all agent playstyles, while these narrative planning approaches often should explicitly fail when they can’t meet the condition set by the user.

Most of the work centered around believable agent behavior that affects individual character behavior in the narrative planning space incorporates two features: characters with differing individual goals and characters who have differing beliefs about the world (Riedl and Young 2010; Teutenberg and Porteous 2013; Ware and Young 2015; Young 2017; Ware and Siler 2021; Shirvani, Ware, and Baker 2022). Approaches where characters have differing goals exist in narrative planning, as often characters may not be ”on the same side” and will pursue opposing goals or will simply have their own intentions. While characters with differing individual goals do behave differently to pursue their goals, these approaches do not consider what actions a character should choose to pursue their individual goals. Since MULTISTYLE is intended for cooperating agents who are all pursuing the same goal, we required an approach where agents could express distinct playstyles without having distinct goals.

Planners which incorporate character belief systems to enhance character believability do generate solution plans where characters take actions based on their belief state, so characters may behave differently from one another. However, characters who perform one action over another because they believe the preconditions of only one action are met aren’t expressing preference, as they are only choosing the action which their beliefs allow them to execute. When the planner is choosing between two actions which the character believes are both executable, the character will not be guided toward an action that best suits the character, as the character has no preferences.

One narrative planner that does incorporate action choice is *Mask* (Bahamón and Young 2017). Characters in *Mask* have personality traits that determine what actions they choose when those actions affect other characters. However, characters do not specifically choose actions based on a preference toward or against the action itself. On the other hand, MULTISTYLE is designed such that agents pursue their own action and proposition preferences when the planner adds steps to the solution plan.

Planner

MULTISTYLE is a centralized planner that generates a single plan for multiple executing agents, so it shares many features of a single-agent planner. The candidate actions of each step of the search process are all actions that could possibly be executed in the world rather than only the actions that a single agent could execute. The planner uses forward heuristic search, constructing a solution plan by starting at the initial state in the problem definition and adding one step at a time until it reaches the goal state. To select which step to add to the plan next, it performs a heuristic evaluation of each potential next step. The heuristic function attempts to guide the planner toward the shortest playstyle-conforming path to the goal, though because the heuristic only provides an approximation, there is no guarantee that the planner's output will be the shortest playstyle-conforming solution plan. Specific information about MULTISTYLE's heuristic can be found in the Heuristic section below.

Definitions

The novel component of the problem definition for the planner is each agent's distinct Playstyle Information set:

Definition 1 (Agent Playstyle Information) *P is the set of propositions in the domain and A is the set of actions in the domain. An agent ϕ's playstyle information L_ϕ is a dictionary of key-value pairs $\{(x \to x_v) \in L_\phi\}$ where $L_\phi(x) = x_v$. x is either a proposition ($x \in P$) or an action ($x \in A$) and $x_v \in \mathbb{R}$ is the corresponding preference value. For propositions ($x \in P$), x_v represents agent ϕ's desire ($x_v > 0$) or aversion ($x_v < 0$) to the atom x being held in the world. For actions ($x \in A$), x_v represents agent phi's desire ($x_v > 0$) or aversion ($x_v < 0$) to perform x.*

The planner uses a multi-agent planning problem with playstyles (PMA-STRIPS). This definition is based on MA-STRIPS (Brafman and Domshlak 2013):

Definition 2 (PMA-STRIPS Problem) *A PMA-STRIPS problem is a septuple $\Pi = \langle P, I, G, \Phi, A, \mathbb{L}, f\rangle$. P is a finite set of t atoms $\{p_j\}_{j=1}^{t}$, $I \subset P$ encodes the initial state of the system, and $G \subset P$ encodes the goal condition. Φ is a set of n agents $\{\phi_i\}_{i=1}^{n}$. A is a set of available actions $\{a_k\}_{k=1}^{o}$. Every action $a \in A$ is a tuple $\langle \phi_i, \text{PRE}(a), \text{ADD}(a), \text{DEL}(a)\rangle$, where ϕ_i is the executor of the action and $\text{PRE}(a)$, $\text{ADD}(a)$, and $\text{DEL}(a)$ are all subsets of P. $\mathbb{L}$ is a set of playstyles $\{L_i\}_{i=1}^{n}$, where L_i is the playstyle information for agent ϕ_i. $f \in \mathbb{N}$ indicates how far past the fixed point the heuristic should extend the relaxed plan graph.*

The primary difference between a MA-STRIPS and a PMA-STRIPS problem is the addition of $\mathbb{L}$, the playstyle set, and f, the heuristic extension value. The value f is further discussed in the Heuristic section. The playstyle set $\mathbb{L}$ allows every agent to have fixed preferences for or against each action the agent could possibly execute and each atomic proposition able to be held in the domain. These preference values are numeric and can be thought of as weights: the higher the preference value, the more the agent prefers the proposition or action, and the lower the preference value, the more the agent dislikes the proposition or action.

Note that there is no addition to the goal condition or any indication of any constraint based on the playstyle. PMA-STRIPS problems should be solved the same way as MA-STRIPS if every item in each agents' playstyle is zero. In that way, agents with sparse preferences will only exhibit different behavior when those preferences are relevant but will otherwise proceed on the shortest path to goal. This is because of a fundamental assumption that the agent will perform actions to pursue their goal and, in a case where the preference of all paths to the goal are equal, the agent will pick the shortest path to reach their goal. The problem itself reflects this by including playstyles for the planning system but not requiring the fulfilment of any specific preference. Instead, the planner is capable of generating metrics on *how well* agents can express their playstyles in a given domain rather than just whether or not they are capable of such expression.

The planner aims to generate a typical solution plan. The definition is included here for completeness, but it is unchanged from a typical solution plan definition:

Definition 3 (Solution Plan) *The solution to a PMA-STRIPS problem is $S = \{a_l\}_{l=1}^{x}$ where $\text{PRE}(a_1) \in I$ and applying a_1 results in an intermediate state $N_1 = \{I \cup \text{ADD}(a_1)\} \setminus \text{DEL}(a_1)$ where $\text{PRE}(a_2) \in N_1$. In turn, applying a_2 to N_1 results in an intermediate state N_2 where $\text{PRE}(a_3) \in N_2$ and so forth until a_x. The state N_x produced by applying a_x to N_{x-1} contains the goal: $G \subset N_x$.*

This solution plan definition contains no preference information and has no condition on playstyles being satisfied in order for the problem to be solved, as the playstyles are not hard constraints but instead are meant to guide the planner.

Description

In addition to its basic planning functionality, MULTISTYLE uses *action pruning*, an approach used in other RPG-based heuristic planners such as FastForward (Hoffmann and Nebel 2001). When the heuristic evaluates a step, it generates a list of helpful actions: actions that the heuristic suggests the planner should consider to reach the goal. Incorporating action pruning in a planner means that, should the evaluated step be added, the planner should consider the helpful actions identified by the evaluation for the next step prior to considering other actions. Normally, this is used to increase the efficiency of the planner, as the planner needs to evaluate fewer potential steps. In MULTISTYLE, however, action pruning is primarily used to make the planner conform to steps the heuristic identified as likely to be on a path that uses high playstyle value actions and propositions in order to reach the goal. This action pruning driven by the heuristic is what ultimately makes MULTISTYLE consider playstyle, as the planner itself does not attempt to evaluate playstyle for potential steps.

The planner uses a typical memoization approach to avoid returning to previously-visited states and rewinds when finding these states, eliminating the previously chosen action from consideration by the planner. If the planner eliminates all of the pruned actions, it will fall back on evaluating and choosing actions beyond the pruned set. This approach means the planner will find a solution should one exist, even if the heuristic returns a set of actions that can't be used to reach the goal, as it will eventually rewind enough to eliminate all pruned actions and move toward the solution.

When the planner invokes the heuristic, it receives the steps in the relaxed plan (where the count of these steps are an approximation of the distance to goal) as well as a value indicating how well the relaxed plan satisfied playstyles. The planner's first priority is to choose the lowest estimated number of steps to goal, as the heuristic is already trying to make these steps conform to the playstyle information. However, if multiple potential next steps in the planner are evaluated with same-length relaxed plans by the heuristic, the planner will choose the step that had a higher playstyle value in its heuristic evaluation.

Consider a case where the planner is considering adding the steps s_1, s_2, and s_3 to the solution plan and the heuristic returns $(0.1, \{a_1, a_2\})$ for s_1, $(0.2, \{a_3, a_4, a_5\})$ for s_2, and $(0.15, \{a_6, a_7\})$ for s_3. The planner will disregard s_2 as it returned more actions (3) than the other options (2), indicating a longer distance to goal. For s_1 and s_3, it will use the highest playstyle evaluation to break the tie, so s_3 will be chosen as the step to add to the plan as its playstyle evaluation was 0.15 compared to s_1's evaluation of 0.1. In this way, solution plans still conform as closely as possible to playstyle even when multiple identical-length paths to goal exist.

The formal definition of the planning algorithm is available in the appendix.

Heuristic

The heuristic controls what actions the planner chooses first to add to the solution plan and sets powerful constraints in the form of action pruning on how the planner should proceed. The heuristic is where the playstyle information included in the problem definition is incorporated. As mentioned before, this is based on RPGPREF, a preference-based heuristic that uses a relaxed plan graph (RPG) to return collections of actions that indicate distance to goal (the count of actions) as well as actions which are likely to lead to the goal which also conform to an agent's preferences. Because MULTISTYLE uses distinct playstyles for each agent, the heuristic needs to incorporate each agent's preferences when building the relaxed plan graph. This makes a centralized planning approach very powerful, as the heuristic can consider all agents' preferences simultaneously when determining the set of actions to return.

In order to run the heuristic, the heuristic needs to be provided with sufficient information to estimate a path to goal. We call this set of information the Evaluation Information.

Definition 4 (Evaluation Information) *The Evaluation Information is a tuple $\langle N, A, G, \mathbb{L}, f\rangle$ where N is the world state to be evaluated, A is the set of all actions in the domain, G is the goal state, $\mathbb{L}$ is the set of playstyle values, and f is a positive integer denoting how far past the fixed point the RPG should be expanded.*

While N, A, and G are required elements for constructing a relaxed plan graph and $\mathbb{L}$ is necessary to incorporate playstyle, f is unique to our heuristic. As the heuristic is not intended to attempt to return minimal sets of actions to approach the goal, it needs to consider more possible causally-linked chains of actions than a typical RPG heuristic. The addition of f in the Evaluation Information allows us to scale the amount of additional paths to goal the heuristic can consider to create a relaxed plan. This results in an addition of $f - 1$ layers past the typical fixed point in RPG generation. Without this extension, some paths to reach the goal may not be considered, and as f gets bigger the heuristic is allowed to consider longer and longer paths to reach the goal. See the appendix for a further explanation and small example of f's purpose in the heuristic.

The heuristic algorithm for constructing the relaxed plan graph is available in Algorithm 1. The relaxed plan graph extraction which creates a relaxed plan from the relaxed plan graph is identical to that of RPGPREF (Lang and Young 2022).

Proposition Preferences

Proposition preferences (what an agent prefers to be true or false in the world state) are factored into the relaxed plan graph as an average over all agents' preferences for each proposition applied to the values of the effects of an action. While PDDL3's proposition preferences (Gerevini and Long 2005) can have additional constraints that specify the duration propositions must hold to satisfy preferences (e.g. *always* or *sometime*), MULTISTYLE instead uses proposition preferences as a means of biasing the heuristic toward actions that have effects containing preferred propositions and away from actions that negate preferred propositions.

An agent ϕ holds a preference value for every atom in the domain $\{\forall p \in P, (p \rightarrow p_v) \in L_\phi\}$. The effect value of each proposition in the heuristic's RPG is composed of all agents' values for p: $\overline{\bigcup_{i=1}^{n} L_i(p)}$. This value is extracted for each effect in line 20 of Algorithm 1. These values are then averaged with all other effects in line 22. As with RPGPREF, during the calculation of the RPG this effect value is averaged with the precondition values and action values to make an overall action value which is set in the action layer (line 23) and also set on the effects in the next proposition layer (line 29 if the effect is not already in the layer or 31 if the new effect value is higher than the existing effect value).

Algorithm 1: Expanding the relaxed planning graph while adding playstyle metadata.

```
**Let:**$lc(\mathbb{K})$ return the max index of the layers in $\mathbb{K}$.
**Let:**$(k \to v)$ be an entry in a dictionary $K$ where $k$ is a key, $v$ is a value, $K(k)$ refers to $v$, and $in(k, K)$ is a boolean function that is true just when the key $k$ exists in $K$.
**Input:**Evaluation information $\langle N, A, G, \mathbb{L}, f\rangle$
**Output:**Planning graph $\mathbb{K}$ with playstyle metadata
    Create graph $\mathbb{K}$
    For initial proposition layer $P0$ in graph $\mathbb{K}$,
        create proposition layer dictionary $K_{P0}$
    Fill $K_{P0}$ such that $\forall n \in N, (n \to 0) \in K_{P0}$
5:  i = 0
    while $lc(\mathbb{K}) < f \vee [\exists p | in(p, K_{pi}) \wedge \neg in(p, K_{Pi-f})]$ do
        $i++$
        Create layer dictionaries $K_{Ai}$ and $K_{Pi}$ in $\mathbb{K}$
        Fill $K_{Pi}$ such that $\forall p \in K_{Pi-1}, (p \to 0) \in K_{Pi}$
10:     for all $a \in A$ do
            Let $\kappa$ be a set of keys for the dictionary $K_{Pi}$
            if $\text{PRE}(a) \subseteq \kappa$ then
                $P = \{\}$
                for all $p \in \text{PRE}(a)$ do
15:                 $P = P \cup K_{Pi-1}(p)$
                end for
                $p_\pi = \overline{P}$
                $E = \{\}$
                for all $e \in \text{EFF}(a)$ do
20:                 $E = E \cup \bigcup_{i=1}^{n} \mathbb{L}_i(e)$
                end for
                $e_\pi = \overline{E}$
                $K_{Ai} = K_{Ai} \cup (a \to \overline{\{p_\pi, e_\pi, \mathbb{L}_{a_\phi}(a)\}})$
                for all $p \in \text{PRE}(a)$ do
25:                 Create edge from $a$ in $K_{Ai}$ to $p$ in $K_{Pi-1}$
                end for
                for all $e \in \text{EFF}(a)$ do
                    if $\nexists e_v | (e \to e_v) \in K_{Pi}$ then
                        $K_{Pi} = K_{Pi} \cup (e \to a_v)$
30:                 else if $a_v > K_{Pi}(e)$ then
                        $K_{Pi}(e) = a_v$
                    end if
                    Create edge from $a$ in $K_{Ai}$ to $e$ in $K_{Pi}$
                end for
35:         end if
        end for
    end while
    if $G \not\subseteq K_{Pi}$ return failure
    return $\mathbb{K}$
```

For a group of agents with distinct proposition preferences, this means the heuristic will prefer neutral actions whose effects average to the highest overall preference value while it avoids neutral actions whose effects average to the lowest overall preference value. In cases where an action's effects are negative but the preference for the action itself is positive, action preference and proposition preference are weighted equally so the average of the two determines how strongly the action is preferred or avoided.

Action Preferences

When the heuristic is evaluating potential actions within the RPG, it only evaluates the preference of the agent who is the executor of the action, not the preference of any other agents. In other words, for an action $a = \langle \phi, \text{PRE}(a), \text{ADD}(a), \text{DEL}(a)\rangle$, the value of action a in agent ϕ's playstyle information is $L_\phi(a)$ which is used as the preference value of that action within the heuristic. In Algorithm 1, line 23 extracts the executing agent's action preference which is incorporated in the overall average for the instance of the action in the RPG. In this way, an agent's actions preferences determine what that agent tends to do and the agent isn't directly influenced by other agents' action preferences. This approach does not allow for agents to have preferences about other agent's actions (e.g. agent ϕ_1 can't prefer that ϕ_2 doesn't fight agent ϕ_1's favorite NPC), nor does it allow for joint actions.

Characterization of Expressivity through Example Playthroughs

In order to illustrate the expressivity of this planner, we invented a specific multi-agent planning problem. The problem was constructed to address requirements for the planner (i.e., requirements 1 and 2 below), to demonstrate diversity and expressivity of solution plans (3 and 4), and to improve the clarity of the features we describe in the generated solution plans (5):

1. Multiple players should be present
2. All players should share the same goal
3. There should be a large variety of different actions available for players to progress toward the goal
4. The minimal length plan to solve the problem should be fairly long
5. Solutions should contain minimal looping and duplicate actions across characters

The premise of the resulting planning problem is a game where three players (named Red, Green, and Blue) awaken as guardians of a fantasy archipelago under threat by a group of ritualists attempting to activate a monolith to cause an apocalypse. The game occurs over three islands where the players progress from island to island by fulfilling certain conditions. For example, on the first island, the players must take a boat from a camp of hostile soldiers. In order to do so, they must either fight the hostile soldiers, requiring them to mine iron, chop trees for wood, and craft swords; or they can sneak past the hostile soldiers, requiring them to solve a puzzle, gather a magical chameleon herb, craft a salve, and camouflage themselves. On the second island, the players must convince a goblin to tell them which island the monolith is on. Upon reaching the third island, the players must disable or destroy the monolith to complete their objective before heading back home.

The domain definition contains 28 objects and 30 actions. A complete description of the domain can be found online[1].

[1]https://zulunko.github.io/research/conference/multistyle.html

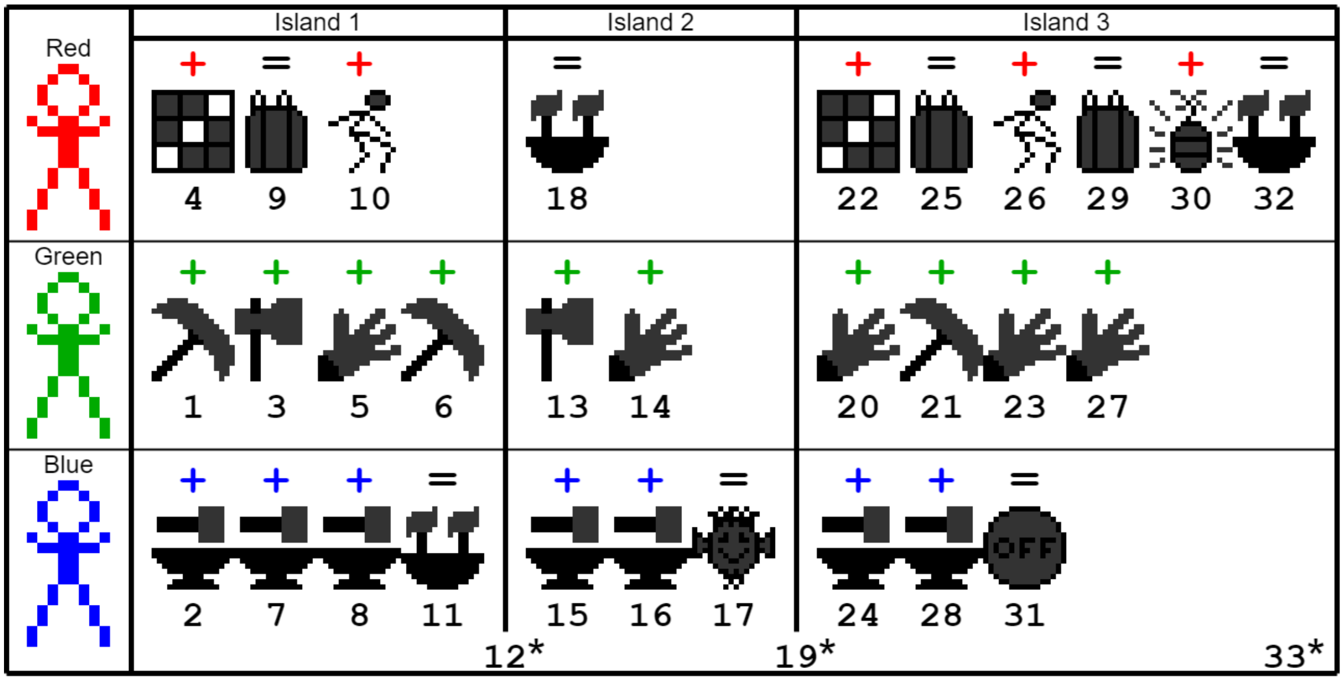

Figure 1: A simple illustration of the actions taken in Playthrough 2. Colored pluses indicate a preference toward the below action, black equals indicate no character has a preference for the action, and the numbers below indicate the step order in the playthrough. Steps 12, 19, and 33 (denoted with an asterisk) are transition actions that simultaneously move every character from one island to the next (step 33 immediately satisfies the goal condition).

Playthrough 1: No Playstyles

A run of the planner without playstyle information results in a fairly expected outcome. The players for each action are not considered, so Red performs the majority of the actions. He mines iron, chops wood, crafts a sword, and attacks the camp on the first island, allowing the team to proceed to the second island. On the second island, Red fights the bandits with his sword, takes their coin, and pays a goblin to tell him where the monolith is. The team proceeds to the third and final island where Blue solves a puzzle to learn how to disable the monolith, then Red fights the ritualists with his sword and disables the monolith before the team returns back home, job complete. While even a simple mediator could be used to prevent Red from performing every action, the planner wouldn't deviate from the path taken to solve the planning problem, as the steps listed above are the shortest path from the initial state to the goal.

Playthrough 2: Harvester, Crafter, Tactician

For the second playthrough, we added the following preferences: Green likes gathering, chopping wood, and mining. Blue likes crafting (there are a number of "craft" related actions in the domain). Red does not like fighting but likes solving puzzles, being sneaky, and putting people to sleep. For all positive preferences $a_v = 1$ and for the negative preference $a_v = -1$.

In this playthrough, the plan gets significantly longer. On the first island, Green mines stone, providing resources for Blue to craft a mortar and pestle. Green chops wood, then Red solves a puzzle to reveal the chameleon herb which Green gathers. Green then mines Iron. Blue crafts the chameleon salve in the mortar and pestle and also crafts a sword. Red equips the chameleon salve and uses it to sneak by the camp. Blue captains the boat to the second island. There, Green chops wood and gathers vines so Blue can craft rope and build a wood house. Blue gives this house as a gift to the goblin in exchange for information about the location of the monolith. Red captains the boat to the third island where Green gathers sleep herb and mines coal. Red solves the puzzle to learn how to disable the monolith. Green gathers sticks. Blue crafts a firesuit out of sticks and coal which Red then equips. Red uses the disguise to sneak past fire elementals to find their closely-guarded firesprites. Green gathers the firesprites and Blue crafts sleep bombs using the herbs and the firesprites. Red equips the sleep bombs and uses them to put the ritualists to sleep, after which Blue disables the monolith and Red captains the boat back home, completing the playthrough. Figure 1 is a simple illustration of the actions in the playthrough organized by the executing player.

This second playthrough is far longer (though is not the longest possible playthrough) because it conforms to the given playstyles for the three players. Additionally, the second playthrough clearly shows the identity of each player as they were defined; Green performs harvesting actions whenever possible, while Blue crafts, Red sneaks, and the player for unpreferred actions is chosen arbitrarily (e.g. captain-

Playthru	Steps	Time	# Nodes	f	Playstyle
1	17	4.81 s	29	1	0
2	33	12.22 s	69	4	0.667
3	32	11.98 s	74	4	0.244

Table 1: Statistics for each playthrough's solution. # Nodes refers to the number of nodes evaluated by the heuristic. Playstyle is the average playstyle value of the steps in the final plan.

ing the boat). None of these playstyle preferences are hard constraints either; if all players dislike a specific action but that action is required to reach the goal, the system will still choose a player to perform the action, as the planner is designed with the assumption that players will attempt to reach the goal even if they don't prefer some actions they necessarily must perform to get there.

Playthrough 3: Tactician, Fighter, Helper

For the third playthrough, we chose opposing preferences within our domain. Red remained as-is, preferring to sneak around, solve puzzles, and put people to sleep. Blue, however, wanted to fight and blow things up. All of these preferences were $a_v = 1$. Green just wanted to help out as best it could, so Green's preference set was a slight preference for anything ($a_v = 0.1$) to prioritize Green for the actions Red and Blue did not have positive preferences for.

In this playthrough, Red and Blue alternate doing the objectives while Green focuses on providing them with the necessary equipment. In the first island, Red first solves the puzzle to unveil the chameleon herb. Green mines stone and gathers the chameleon herb before crafting a mortar and pestle and a chameleon salve. Green then chops wood and mines iron to craft a sword. Red equips the chameleon salve and uses it to sneak by the camp to get the soldiers' boat and Green takes the helm. Blue equips the sword Green crafted and the group moves to the second island.

In the second island, Green gathers flint and chops wood to craft a torch which Blue then equips. Blue uses this torch to threaten the goblin who surrenders the information about the location of the monolith. Green then captains the boat to the third island.

On the third island, Blue fights fire elementals to find the firesprites which Green then gathers. Green mines stone and then uses it with the firesprites to craft demolition bombs. Green then gathers sleep herbs and crafts sleep bombs. Red equips the sleep bombs and sleeps the ritualists, then Blue equips the demolition bombs and blows up the monolith. Green captains the boat back home.

On the first island, Red got to do what it wanted while Blue did practically nothing aside from equipping a sword. On the second island, Blue gets to threaten the goblin but Red does nothing. On the third island, both Blue and Red get to satisfy their playstyles to some extent: Blue fights elementals and blows up the monolith and Red sleep bombs the ritualists. In all islands, Green performs any actions needed to allow Red and Blue to perform their actions and captains the boat when the group is moving between islands.

Additional Statistics

Table 1 shows data about the performance of the planner for each playthrough. The preference data clearly causes the planner to take longer to complete, but this extra time is explained by the significantly longer solution plans. This higher step count is not only expected but desired for the system's output (as the purpose of the planner is to diverge from shortest-length solutions by prioritizing playstyle), so we consider the additional time needed to calculate the solution plan to be acceptable, particularly as there is no current plan to incorporate this in a system that would rely on solution plans being generated under time constraints. The nodes evaluated may seem fairly low, but this is a product of the action pruning used by the system: since the previous step provides a list of helpful actions, the heuristic is frequently only evaluating two or three actions for each step past the first.

$f = 1$ is the default f value for a planner finding the shortest plan and extending f beyond 1 in Playthrough 1 results in the exact same solution being generated in an increased time. $f = 4$ is sufficient to include all possible paths to goal in this example domain and reducing f below 4 does result in different solution plans for Playthroughs 2 and 3. For example, if f is set to 1, the steps through the third island in Playthroughs 2 and 3 match Playthrough 1 as Playthrough 1 takes the shortest path through the third island (though different characters execute the actions as determined by playstyle).

The playstyle values have to be understood in the context of the playstyle sets provided for each playthrough. In both Playthrough 2 and 3, the planner returned the solution plan with the maximum possible average playstyle in the domain given the playstyle sets being used. Because the players in Playthrough 2 had strong and significantly different playstyles, the overall average value was high. In Playthrough 3, Red and Blue had strong playstyles that conflicted somewhat and prevented them from working together while Green had very slight playstyle values, so the resulting playstyle average was lower.

Discussion and Future Work

Our intent in developing this planner is to eventually visualize the output for game level designers to help illuminate how teams of players may play through their levels. There are some possible inferences a designer might make about their game levels when given information from MULTISTYLE. For example, in the second playthrough illustrated in Figure 1, the Red player does not have any positive playstyle actions to perform in the second island. A designer could possibly see this as a flaw in their level and add something that gives a Red-style player something to do in the second island. In the third playthrough, the designer may want to add to the domain to enable both Red (the Tactician) and Blue (the Fighter) to act in their preferred ways on each island, as those two agents don't get to satisfy their playstyle on each island and the playstyle average value is low as a result. In all playthroughs, there are fewer actions in the second island than the other islands which may be desirable or undesirable to a designer and could prompt them to improve their level.

If a designer viewed many playthroughs, they could identify places in the level that players will likely never access or paths the players seem to always take, allowing them to modify the domain accordingly. For example, there are bandits on the second island that a player can fight to find gold coins which can be gathered to pay the goblin for information containing the location of the monolith. This path was used Playthrough 1 (no playstyle). If an analysis showed a designer that few to no playthroughs contained that path, they could change the path to be more appealing or eliminate it entirely. There are many potential insights that designers could glean from the output of this system if it were presented in an understandable way.

We see four primary areas where this work can be extended. First, we would like to incorporate the system as a level design tool in an AI-assisted game level editor (similar in implementation to other AI-assisted level design tools like Tanagra (Smith, Whitehead, and Mateas 2010), work related to Procedural Personas (Liapis et al. 2015), or the AI playtesting visualization work by Agarwal et al. (Agarwal et al. 2020)). Instead of directly providing suggestions for how a level designer might change their levels, the tool would provide designers with an analysis about how their games may be played. While a designer may be able to predict what one group of players will do, it may be very hard for them to directly compare what they believe many different groups of players would do in their levels. Additionally, the less linear a game level becomes, the harder it gets to predict how different types of players will experience each level. Ideally, our system would help alleviate this problem by providing the designer with additional information which they can then use to make more informed decisions about modifying their levels. This future work is UI-focused, as it requires designers to be presented with multiple different playthroughs at once so they can compare and understand them within a level editing environment.

The second area of future work involves deriving playstyle values from, for example, data mining real players. Given a successful game or game series, a designer may be tasked with making new levels for sequels or expansions. The designer may already have a wealth of player data available to use but may not have an easy way of translating that player data to distinct playstyles for our system. Clustering players into types would allow a designer to more easily understand what sort of players play their games. This data could be directly given to the planner as a playstyle set, allowing the planner to automatically provide representative playthroughs without any manual playstyle definition by the designer. To aid in this effort, the playthroughs generated by MULTISTYLE could be tested against real players in a variety of domains. While real players are unlikely to behave exactly as generated in the planner's solution plans, proving that MULTISTYLE has some degree of accuracy in predicting real players would be valuable.

Third, MULTISTYLE assumes that all players have complete knowledge of the domain (and thereby the freedom to choose exactly what they want to do to complete the goal). This is unrealistic, as certainly the first time a player begins a game level they may not be able to immediately understand the sequences of actions available that they can take to reach the goal. One avenue of future work involves incorporating exploratory behavior to help understand how players with no knowledge of a game level might approach completing a level given that each player has a distinct playstyle.

Finally, we will look to extend the MULTISTYLE planner to address the generation of multi-character narrative plans. For the field of narrative planning, approaches like MULTISTYLE where agents can have distinct preferences about actions could result in more believable characters. While some narrative planners have already incorporated some character traits to express character personality (Bahamón and Young 2017; Shirvani, Ware, and Baker 2022), the character behavior has been based on the impact of their actions, not on whether the character enjoys performing the action itself. Because real people have certain activities that they enjoy more than others, one could reasonably assume that incorporating action preferences in a narrative planner would result in characters who seem more humanlike.

Conclusion

In this paper, we presented a preference-based multi-agent planner named MULTISTYLE. This planner is capable of generating solution plans that express agent individuality by incorporating agent-specific action and proposition preferences. These preferences, collectively referred to as a playstyle, are used by the heuristic to guide the planner toward solution plans that contain actions and world states that match each agent's playstyle. We included an example domain and multiple playthroughs within that domain to present the variety of plans that could be produced by solely varying the preference set for agents. These playthroughs clearly show that MULTISTYLE disregards shortest-path solutions in favor of those that conform to defined playstyles for each agent within the domain.

While MULTISTYLE sits adjacent to the fields of preference-based planning, multi-agent planning, and narrative planning, it falls in an unexplored area between the three. Preference-based multi-agent planners are an unrepresented class of algorithms in research and, while narrative planners tend to differentiate characters by having them pursue different goals, no narrative planners directly incorporate characters' action preferences. The increased expressive range available to MULTISTYLE sets the stage for the development of tools that allow designers to easily understand how teams of players may play through their levels even when each player within the team has a distinct playstyle.

A Heuristic's f Value

The value f is important for allowing the heuristic to find longer action chains that lead to the goal. This increased capability is beneficial in that it could help agents better express their playstyles.

As a brief example of why f allows for these longer action chains, if the action sequence A, B, C results in a state that allows the agent to perform the action G to complete

Algorithm 2: MULTISTYLE's planning algorithm.

```
Input:A PMA-STRIPS problem $\langle P, I, G, \Phi, A, \mathbb{L}, f \rangle$
Output:A solution plan S
    $S = \{\}$ // Solution plan
    $N = \{I\}$ // World states (memoization)
    $R = \{\}$ // Action pruning lists
    while $G \nsubseteq N_{|N|}$ do
5:      // Memoization:
        if $N_{|N|} \in \{N_1, N_2, ..., N_{|N|-1}\}$ as $N_x$ then
            $R_{x-1} = R_{x-1} \setminus S_x$
            $N = \{N_1, N_2, ..., N_x\}$
            $R = \{R_1, R_2, ..., R_{x-1}\}$
10:         $S = \{S_1, S_2, ..., S_{x-1}\}$
        end if
        $E = \{\}$ // Executable actions
        if $R_{|R|} \neq \emptyset$ then
            for all $a \in R_{|R|}$ do
15:             if $\text{PRE}(a) \in N_{|N|}$ then
                    $E = E \cup a$
                end if
            end for
        else
20:         for all $a \in A$ do
                if $\text{PRE}(a) \in N_{|N|}$ then
                    $E = E \cup a$
                end if
            end for
25:     end if
        if $|E| = 0$ then
            $R_{|R|} = R_{|R|} \setminus S_{|S|}$
            $R = R \setminus R_{|R|}$
            $N = N \setminus N_{|N|}$
30:         $S = S \setminus S_{|S|}$
            continue
        end if
        $v_b = -1$
        $\Gamma_b = -1$
35:     $a_b = null$
        $C_b = \{\}$
        for all $a \in E$ do
            $C = \{N_{|N|} \cup \text{ADD}(a)\} \setminus \text{DEL}(a)$
            $\langle v, \Gamma \rangle = Heuristic(C, A, G, \mathbb{L}, f)$
40:         if $v_b = -1 \vee v < v_b \vee (v = v_b \wedge \Gamma > \Gamma_b)$ then
                $v_b = v$
                $\Gamma_b = \Gamma$
                $a_b = a$
                $C_b = C$
45:         end if
        end for
        $R = R \cup \Gamma_b$
        $N = N \cup C_b$
        $S = S \cup a_b$
50: end while
    return $S$
```

the goal, a valid path to reach the goal is A, B, C, G. However, if the action sequence D, E results in the exact same world state as A, B, C, expanding only to the fixed point will result in a relaxed plan graph of three action layers: the first with A, D, the second with A, D, B, E, and the third with A, D, B, E, C, G. Because C has the same effects on the world as E, the third proposition layer (after the third action layer) will be identical to the second proposition layer, meaning the RPG has hit its fixed point. If we assume we've stopped at the fixed point, when extracting the relaxed plan we will select G from the third layer, look for lower-layer actions that provide G's preconditions and select E from the second layer (because C does not exist in the second layer), and then select D from the first layer. In order to consider the chain A, B, C, G, the RPG must be expanded one layer further so G exists at least one layer above C.

In typical RPG construction, preventing longer action chains is intended and desirable behavior: the heuristic is more efficient if it has fewer layers and doesn't need to consider all paths to the goal. In our heuristic, however, we want to consider potential paths which are lengthier than the shortest-length path, so f determines how long those potential paths can be. Reducing f makes the heuristic finish its evaluation faster but possibly disregard longer action chains and produce shorter solutions, while increasing f makes the heuristic take longer but possibly include longer action chains and produce longer solutions. f should therefore be tuned both based on the desired playstyle-adherence of the user as well as the length of action chains within the domain.

B Planner Algorithm

A full planner algorithm can be found in Algorithm 2. S, N, and R are ordered and are the solution plan steps, the world state at each step (used for memoization), and the pruned action list (used for action pruning). Because N contains the initial state and a state after every step, it has a length of one more than the other sets. Note that R_x contains the pruned action list to be considered for the step after S_x; e.g. R_1 limits the planner's evaluation for which step to choose as S_2.

References

Agarwal, S.; Herrmann, C.; Wallner, G.; and Beck, F. 2020. Visualizing AI Playtesting Data of 2D Side-scrolling Games. In *2020 IEEE Conference on Games (CoG)*, 572–575.

Bahamón, J.; and Young, R. 2017. An Empirical Evaluation of a Generative Method for the Expression of Personality Traits through Action Choice. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 13(1).

Baier, J. A.; Bacchus, F.; and McIlraith, S. A. 2009. A heuristic search approach to planning with temporally extended preferences. *Artificial Intelligence*, 173(5): 593–618. Advances in Automated Plan Generation.

Bartle, R. 1996. Hearts, Clubs, Diamonds, Spades: Players Who Suit MUDs. https://mud.co.uk/richard/hcds.htm. Accessed: 2023-08-24.

Bateman, C.; Lowenhaupt, R.; and Nacke, L. 2011. Player Typology in Theory and Practice. In *DiGRA &#3911 - Proceedings of the 2011 DiGRA International Conference: Think Design Play*. DiGRA/Utrecht School of the Arts. ISBN ISSN 2342-9666.

Brafman, R. I.; and Domshlak, C. 2013. On the complexity of planning for agent teams and its implications for single agent planning. *Artificial Intelligence*, 198: 52–71.

Coles, A.; and Coles, A. 2011. LPRPG-P: Relaxed plan heuristics for planning with preferences. In *Twenty-First International Conference on Automated Planning and Scheduling*.

Gerevini, A.; and Long, D. 2005. Plan Constraints and Preferences in PDDL3 The Language of the Fifth International Planning Competition. *ICAPS 2006*.

Gerevini, A. E.; Lipovetzky, N.; Percassi, F.; Saetti, A.; and Serina, I. 2019. Best-first width search for multi agent privacy-preserving planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, 163–171.

Hamari, J.; and Tuunanen, J. 2014. Player Types: A Meta-synthesis. *Transactions of the Digital Games Research Association*, 1: 29–53.

Hoffmann, J.; and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research*, 14: 253–302.

Kallio, K. P.; Mäyrä, F.; and Kaipainen, K. 2011. At Least Nine Ways to Play: Approaching Gamer Mentalities. *Games and Culture*, 6(4): 327–353.

Lang, E. W.; and Young, R. M. 2022. RPGPref: A Planning Heuristic That Uses Playstyle Preferences to Model Player Action and Choice. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 18, 129–136.

Laws, R. 2002. *Robin's Laws of Good Game Mastering*. Steve Jackson Games.

Liapis, A.; Holmgård, C.; Yannakakis, G. N.; and Togelius, J. 2015. Procedural personas as critics for dungeon generation. In *Applications of Evolutionary Computation: 18th European Conference, EvoApplications 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings 18*, 331–343. Springer.

Maliah, S.; Shani, G.; and Stern, R. 2017. Collaborative privacy preserving multi-agent planning: Planners and heuristics. *Autonomous agents and multi-agent systems*, 31: 493–530.

Mora, A.; Tondello, G. F.; Calvet, L.; González, C.; Arnedo-Moreno, J.; and Nacke, L. E. 2019. The Quest for a Better Tailoring of Gameful Design: An Analysis of Player Type Preferences. In *Proceedings of the XX International Conference on Human Computer Interaction*, Interacción '19. New York, NY, USA: Association for Computing Machinery. ISBN 9781450371766.

Nissim, R.; and Brafman, R. 2014. Distributed heuristic forward search for multi-agent planning. *Journal of Artificial Intelligence Research*, 51: 293–332.

Riedl, M. O.; and Young, R. M. 2010. Narrative Planning: Balancing Plot and Character. *Journal of Artificial Intelligence Research*, 39: 217–268.

Shirvani, A.; Ware, S. G.; and Baker, L. J. 2022. Personality and Emotion in Strong-Story Narrative Planning. *IEEE Transactions on Games*.

Smith, G.; Whitehead, J.; and Mateas, M. 2010. Tanagra: A Mixed-Initiative Level Design Tool. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, 209–216. New York, NY, USA: Association for Computing Machinery. ISBN 9781605589374.

Teutenberg, J.; and Porteous, J. 2013. Efficient intent-based narrative generation using multiple planning agents. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 603–610.

Torreño, A.; Onaindia, E.; Komenda, A.; and Štolba, M. 2017. Cooperative Multi-Agent Planning: A Survey. *ACM Comput. Surv.*, 50(6).

Torreno, A.; Onaindia, E.; and Sapena, O. 2014. FMAP: Distributed cooperative multi-agent planning. *Applied Intelligence*, 41: 606–626.

Tseng, F.-C. 2011. Segmenting online gamers by motivation. *Expert Systems with Applications*, 38(6): 7693–7697.

Vahlo, J.; Kaakinen, J. K.; Holm, S. K.; and Koponen, A. 2017. Digital Game Dynamics Preferences and Player Types. *Journal of Computer-Mediated Communication*, 22(2): 88–103.

Ware, S. G.; and Siler, C. 2021. Sabre: A Narrative Planner Supporting Intention and Deep Theory of Mind. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 17(1): 99–106.

Ware, S. G.; and Young, R. M. 2015. Intentionality and conflict in The Best Laid Plans interactive narrative virtual environment. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(4): 402–411.

Ye, D.; Zhu, T.; Shen, S.; Zhou, W.; and Philip, S. Y. 2020. Differentially private multi-agent planning for logistic-like problems. *IEEE transactions on dependable and secure computing*, 19(2): 1212–1226.

Young, R. M. 2017. Sketching a Generative Model of Intention Management for Characters in Stories: Adding Intention Management to a Belief-Driven Story Planning Algorithm. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 13(1).