Proceedings of the Eighteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment
(AIIDE 2022)

RPGPref: A Planning Heuristic that Uses Playstyle Preferences to Model Player Action and Choice

Eric W. Lang[1], **R. Michael Young**[1,2]
[1]School of Computing, University of Utah
[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

Recent work extending planning algorithms that reason about action and change has been successful at supporting game design, player modeling, and story generation. Incorporating agent preferences over actions and propositions into a planning process allows for a more accurate prediction of what a human might do when solving a problem like playing through a game level. This paper presents the preference-based planning heuristic RPGPREF which uses relaxed plan graphs (RPGs) and preference sets to guide a planner toward a preference-conforming path to its goal. A human subjects evaluation confirms that RPGPREF successfully guides the planning process toward solution plans that recognizably match and differentiate player playstyles.

Introduction

Game designers working across many genres will often design game levels that require players to take long sequences of actions to achieve goals in ways that require those sequences to respect complicated causal dependencies between actions. In games with many choices, anticipating what a certain type of player might do can be extremely difficult, but such predictions could help a designer find ways to better engage that type of player. Planning algorithms are powerful tools capable of producing multi-step solutions to fairly complex problems where causal dependency and goal achievement are central.

Typically, planning research has revolved around incorporating more complex features into planning or reducing the amount of time that a planner takes to generate an optimal (shortest-length) solution. However, planners which solely produce shortest-length plans may not match how a human would solve the same problem. Not only can multiple shortest-length plans exist, but also humans don't always behave in ways that an algorithm would consider optimal. Recent work has used planning to give game designers more visibility into the planning process (Pizzi et al. 2021) or generate multiple plans and select one that matches a player model (Ramirez and Bulitko 2015). However, in order to more accurately predict human behavior within the planning process, the human's preferences about *how* they solve a problem must be taken into account. While a preference-aware planner might be used to more accurately predict human behavior in a variety of real-world scenarios, video games provide the perfect testbed for these planners. Game and level designers often attempt to accommodate multiple player types in their games and a preference-aware planner could help a designer in this process. In this paper, we often refer to the way a player prefers to play a game as their *playstyle*; a collection of data that defines what a certain type of player wants to do or how they want the game world to be.

This paper presents a search heuristic called RPGPREF – roughly based on Hoffman and Nebel's FastForward heuristic (Hoffmann and Nebel 2001) – that is used to drive a HSP algorithm in its search to construct a sequence of actions to achieve a desired goal state. RPGPREF uses preference values to bias the construction and analysis of a relaxed plan graph (RPG) built by HSP planners to select the next best actions to add to the suffix of a plan. These biases guide a planner to select steps which either the player prefers or have effects which the player prefers. Similarly, it pushes the relaxed plan extraction away from steps which are undesirable to the player. This heuristic-based approach guides the planner to appropriate playstyle-conforming paths by both minimizing the steps in the plan while maximizing the playstyle value of the plan.

Related Work

Preference-based planning has been an active research area for well over a decade. These preference-based planners typically try to solve a planning problem that both minimizes cost and maximizes utility or desired plan properties. Initially, these planners primarily targeted a specific preferred end state, but other heuristic-based methods such as HPLAN-P (Baier, Bacchus, and McIlraith 2009) and LPRPG-P (Coles and Coles 2011) can target specific propositions in the planning process. These approaches, like RPGPREF, use an enriched relaxed plan graph, but they only allow for proposition preferences rather than action preferences and their preferences are more rigid than the preference values used in this work. Because gameplay necessarily involves player agency within the world, the ability to incorporate action preference is vital to producing accurate plans.

Some narrative planners (e.g. IPOCL (Riedl and Young 2010), CPOCL (Ware and Young 2011), Glaive (Ware and Young 2014), IMPRACTICAL (Teutenberg and Porteous 2013), HEADSPACEX (Young 2017)) implement character choices using a belief-desire-intention (BDI) model based on Bratman's model of practical reasoning (Bratman 1991). In these systems, characters can have their own goals and so they perform actions to pursue those goals but do not incorporate preferences in action selection.

ACONF is a narrative generation system focusing on generating believable characters by expressing character traits (Riedl and Young 2003). While they mention a heuristic function which captures a character's action preferences, they do not describe how such a heuristic would work within their system.

Mask is a planner that incorporates character personality into the action choices they make in a narrative (Bahamón and Young 2017). This approach determines action choice by analyzing the impact of actions on other characters' emotions rather than explicit enumeration of character preferences. The work by Rivera-Villicana et al. (2018) on using BDI for player modeling touches on a player's goal choice in the context of an interactive narrative. Their model has exploration as a type of action choice where the player may choose to perform actions which don't lead directly to the goal in order to gather information but do not consider alternate ways the player may approach their goal once they have all the necessary information.

Heuristic Description

In this section, we provide a detailed description of a heuristic-driven search process for constructing plans. Additional background describing the elements of our planning knowledge representation appears in the appendix.

Heuristic search planners typically perform depth-first search without backtracking to generate a plan by using a heuristic to guide the process. This heuristic determines, at each step of the planning process, which action should be selected as the next step in the plan. The planner gives the heuristic information about each next possible step being considered and the heuristic returns some evaluation of those steps. Because of this, the heuristic has the power to guide the planner toward desired characteristics. However, because the heuristic must practically be far faster than the planner itself, all of the returned values are estimates. In many planning contexts, the desired plan achieves a goal state in as few steps as possible. In these cases, the heuristic is optimized to estimate the number of steps to the goal, allowing the planner to select the next step which has the estimated shortest distance to the goal. By doing so until the goal is reached, the planner can efficiently find a plan that is reasonably optimal (depending on the accuracy of the heuristic).

Approaches that use relaxed plan graphs such as the one presented in this paper are powerful because, when evaluating which step the planner should choose next, they attempt to look forward in the planning process by building a relaxed graph of possible future steps and extracting a collection of actions that might be used to reach the goal. In a system such as FastForward, the number of actions in this collection is a relatively accurate estimate of the distance from the evaluated step to the goal. On the other hand, by specifically targeting actions and propositions that are preferred in this extraction process, RPGPREF can bias the relaxed plan extraction to select actions which are both productive (i.e. they approach the goal) but also express the preferences defined by the user. This allows the heuristic to approximate the distance to the goal while also attempting to conform to the given preferences. The planning process selects each next step that has the shortest estimated preference-conforming path to the goal.

For this preference-aware heuristic, we consider *playstyle* to be a simulated player's preference for specific actions or states in the world. We incorporate playstyle preferences as arbitrary float values assigned to specific propositions and actions that tell the system how well the specific action matches a playstyle or how likely it is that a player would pursue or avoid a specific proposition.[1] Effectively, this means that a final plan can be evaluated for a specific playstyle as a function of the values of all actions in that plan and the values of all world states it progresses through calculated by analyzing the propositions that hold for each individual state.

Definition 1 (Proposition Preference) *A proposition preference is a tuple $\langle p, v\rangle \in X_p$ which contains a proposition p and a float value v indicating the preference value for p in some playstyle.*

Definition 2 (Action Preference) *An action preference is a tuple $\langle a, v\rangle \in X_a$ which contains an action a and a float value v indicating a preference value for a in some playstyle.*

Definition 3 (Playstyle) *A playstyle $\mathcal{P}$ is a set of proposition and action preferences $\langle X_p, X_a\rangle$ that collectively represents the preferences of a certain type of player.*

RPGPREF expands its relaxed plan graphs all the way to the *fixed point* – the first repeated proposition layer in the relaxed plan graph where no new actions will be available in the next layer – because the algorithm optimizes for the actions chosen and the world states traversed in our plan rather than the length of the plan. By expanding all the way to the fixed point, we ensure that all actions and propositions that are reachable by the state being evaluated are represented in the RPG. Our resulting relaxed plan, therefore, is produced by considering all possible steps to reach the goal.

Our relaxed plan extraction algorithm starts at the fixed point, selects all goals, and works backwards selecting actions to produce those goals, preferring desired actions and actions which produce desired propositions. The algorithm described here skips layers only when such a skip does not result in a loss of playstyle preference value. This is distinct from a typical RPG approach in a few ways, often losing accuracy on measuring the shortest path to goal but potentially

[1] This approach is hand-crafted in this paper, though one might consider more data-driven approaches (e.g., Monte Carlo tree search (Silver et al. 2016)) that could be used to generate action sequences driven by learned preference values.

gaining accuracy on measuring a more realistic and representative path to goal for a given type of player.

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

Input: Evaluation information $\langle A, s, g, \mathcal{P} \rangle$
Output: Planning graph G with playstyle metadata

```
    Create planning graph G
    Create proposition layer L_P0 in G
    for each proposition p in s do
        Add p to L_P0, p_v = 0
 5: end for
    fixedpoint = false
    i = 1
    while fixedpoint = false do
        Create action layer L_Ai in G
10:     Create proposition layer L_Pi in G
        for each proposition p in L_Pi-1 do
            Duplicate p as p', p'_v = p_v
            L_Pi = L_Pi ∪ p'
        end for
15:     for each action a in A do
            if a's preconditions exist in L_Pi-1 then
                L_Ai = L_Ai ∪ a
                for each precondition p of a do
                    Connect a in L_Ai to p in L_Pi-1
20:             end for
                p_avg = avg a's precondition values in L_Pi-1
                e_avg = avg a's effect values in P
                a_v = avg(p_avg, e_avg, P_a)
                for each effect e of a do
25:                 if e ∉ L_Pi then
                        L_Pi = L_Pi ∪ e, e_v = a_v
                    else if a_v > e_v then
                        e_v = a_v
                    end if
30:                 Connect e in L_Pi to a in L_Ai
                end for
            end if
        end for
        if L_Pi = L_Pi-1 then
35:         fixedpoint = true
            if g ⊈ L_Pi return failure
        end if
        i++
    end while
40: return G
```

A baseline assumption in this algorithm is that the heuristic does not need to incorporate actions that do not lead the agent toward the goal in some way. This eliminates frivolous actions: the relaxed plan won't include an attack, for example, unless the attack action either fulfills a goal or fulfills a precondition of some series of actions which fulfill a goal. This does mean that the algorithm posed here is only useful in an agency-rich environment; the player must be able to make choices that move them toward their goal and there must not be only one way to proceed to the goal, as otherwise FastForward would simply be a more efficient version of this algorithm.

The RPGPREF heuristic calculation takes in a state information tuple that contains all information the heuristic calculator needs to know about the domain and the goal as well as the playstyle that is being calculated for.

Definition 4 (Evaluation Information) *The evaluation information is a tuple $\langle A, s, g, \mathcal{P} \rangle$ that contains all actions in the domain A, the world state to be evaluated s, the goal g, and a playstyle $\mathcal{P}$.*

Expanding the Relaxed Plan Graph

Algorithm 1 shows our method to expand the relaxed plan graph for use in relaxed plan extraction. This tightly follows typical relaxed plan expansion algorithms, with a few key differences.

First, the RPG is expanded all the way to the fixed point, ensuring that all possible paths to goal are represented in the RPG. If the goal is not represented in the final layer, it is unreachable.

Second, a playstyle value is assigned to every node. The algorithm carries playstyle values forward to each proceeding layer. The initial propositions have no playstyle value, as the actions in the relaxed plan should not be deprioritized for relying on existing propositions. This is also true in the rest of the algorithm: as values propagate through the graph, the precondition values are only affected by the actions and effects of the actions that may have produced them, not by their value within the playstyle. In this way, we ensure that we bias action selection based on actions and their outcomes and not based on their requirements.

In each action layer, the actions have a value which is calculated as an average of averages $avg(p_{avg}, e_{avg}, \mathcal{P}_a)$. An action's playstyle value is calculated from three parts: its precondition values (how much the actions leading to this action have matched the player's playstyle), its actual value in the playstyle (how much this specific action matches the player's playstyle), and its effect values (how much the player likes or dislikes this action's effects on the world state). We assume these three have equal weight for the purposes of our heuristic. Any action or effect that does not have a playstyle value is assigned a zero for the purpose of averaging.

Any given action may have a value in the playstyle, and this value occupies the $\mathcal{P}_a$ slot when calculating $a_v = avg(p_{avg}, e_{avg}, \mathcal{P}_a)$. The effect values e_{avg} are the average of the effect propositions (excluding unvalued propositions) within the set of proposition preferences X_p in the playstyle $\mathcal{P}$. The precondition values are inherited from the action that produced them (as described next) or are unvalued. Once a_v is calculated, it is assigned as the value of a but also as the value of every effect of a in the next proposition layer. The values that are assigned to the effects of a are then used in the next action layer's calculations as precondition values.

Because any given proposition in a proposition layer can be produced by multiple different actions in the preceding action layer, we set the value of a proposition to the highest value it received from actions which produced it in the pre-

Algorithm 2: Calculating heuristic values

Input: Planning graph with playstyle metadata & goal $\langle G, g \rangle$
Output: Playstyle matching value v and pruned action set A

i = final layer index of the planning graph $len(G)$
g_{Pi} = tuples of each goal g in L_{Pi} and their values
$v = avg(g_{Pi}$ values$)$
$A = \{\}$
while $i > 0$ **do**
  $S = \{\}$
  for each goal proposition p_g in g_{Pi} from highest value to lowest value **do**
    if $p_g \in S$ **then**
      continue
    end if
    a = highest value action that provides p_g
    $A = A \cup a$
    for each effect e of a **do**
      $S = S \cup e$
    end for
    for each precondition p of a **do**
      $c = p_v$ in L_{Pi-1}
      $j = i - 1$
      while $j >= 0$ **do**
        if $p \notin L_{Pj}$ or (p_v in L_{Pj}) $< c$ **then**
          $g_{Pj+1} = g_{Pj+1} \cup p$
          break
        end if
        $j - -$
      end while
    end for
  end for
  $i - -$
end while
return v, A

vious layer. In the algorithm, this means that the effect of an action is only assigned the action's value in the next proposition layer if the corresponding proposition doesn't already have a greater value assigned by some other action.

At the point where the plan graph constructs its last layer (which is at the fixed point), the RPG is complete and the relaxed plan is ready to be extracted. The values that are assigned to the goals in the final proposition layer approximate how well the relaxed plan will match the given playstyle.

Extracting the Relaxed Plan

Algorithm 2 shows our method of extracting the relaxed plan and thereby calculating the heuristic value from our calculated plan graph from Algorithm 1. All of the information needed is contained in the RPG and the set of goals to be achieved. The main differences in our relaxed plan extraction compared to FastForward are in step selection and skipping proposition layers for subgoals.

As with FastForward, our relaxed plan extraction starts at the last layer. For each goal, the algorithm searches backwards in the planning graph for the point where that goal's proposition no longer exists *or* the playstyle value of the proposition drops. Upon finding either of these conditions, it assigns that goal to the previously-searched layer (i.e. before the value is dropped or while the goal still exists as a proposition). While FastForward solely searches back through the layers to find the first layer that a goal appeared, our algorithm also stops its search if it finds that the goal drops in playstyle value. This way, we ensure that we always select a layer where the previous actions that led to this proposition were as high-value as possible but still reduce the relaxed plan length as much as we can. We use this exact same process for subgoals (preconditions of actions which we add as steps), resulting in a relaxed plan that is as short as possible while still retaining maximum value.

For step selection, then, we start at the last layer that contains a goal starting with the highest-value goal and select the highest-value action that provides that goal (which should have the same value as the goal proposition). We then add that action's preconditions as subgoals using the process described in the previous paragraph, starting with the proposition layer immediately preceding the selected action. This continues in a loop, always prioritizing the final layer that still contains an unresolved goal and then prioritizing the goals in order of descending playstyle value.

Once all goals are met by the relaxed plan, the selected actions A are the steps in the relaxed plan for the heuristic calculation. As is typical in RPG-based approaches, the heuristic value used is simply the number of steps selected $len(A)$ and, if the step being evaluated is chosen by the planner, the actions in A will be the actions first considered for the proceeding step. The playstyle value v of the final goals is currently only used to prioritize the selection of steps with equal length estimates.

Simple Example: Lights

In our example, we define a simple domain where there are two rooms: room 1 and room 2. The goal in our example is for the lights to be on. One possible solution, referred to as path 1, involves the agent flipping a switch in room 1, thus turning on the lights and immediately meeting the goal. The other solution, path 2, involves the agent picking up a key in room 2 and using the key to turn a switch that turns on the lights. Path 2 is clearly more total actions (2 instead of 1) once the agent is in the room. We will assume there is an entrance that allows the agent to enter either room, but both rooms are connected to each other.

First, let's consider the case of a FastForward heuristic. At the start of the planning process, the planner evaluates starting by entering room 1 versus entering room 2. Entering room 1 will show one action to goal while entering room 2 will show two actions to goal, so the planner will plan for room 1 and proceed from there to select path 1, flipping the switch and turning on the lights.

Next, let's consider a case where the agent has a playstyle that makes them prefer room 2. This could be, for example, a preference for the true proposition `(at agent room2)`, but it could also be a preference for using the key to turn on the lights or a preference against a proposition or action for path 1. When evaluating the move to room 2, RPG-PREF behaves much like the FastForward heuristic: it re-

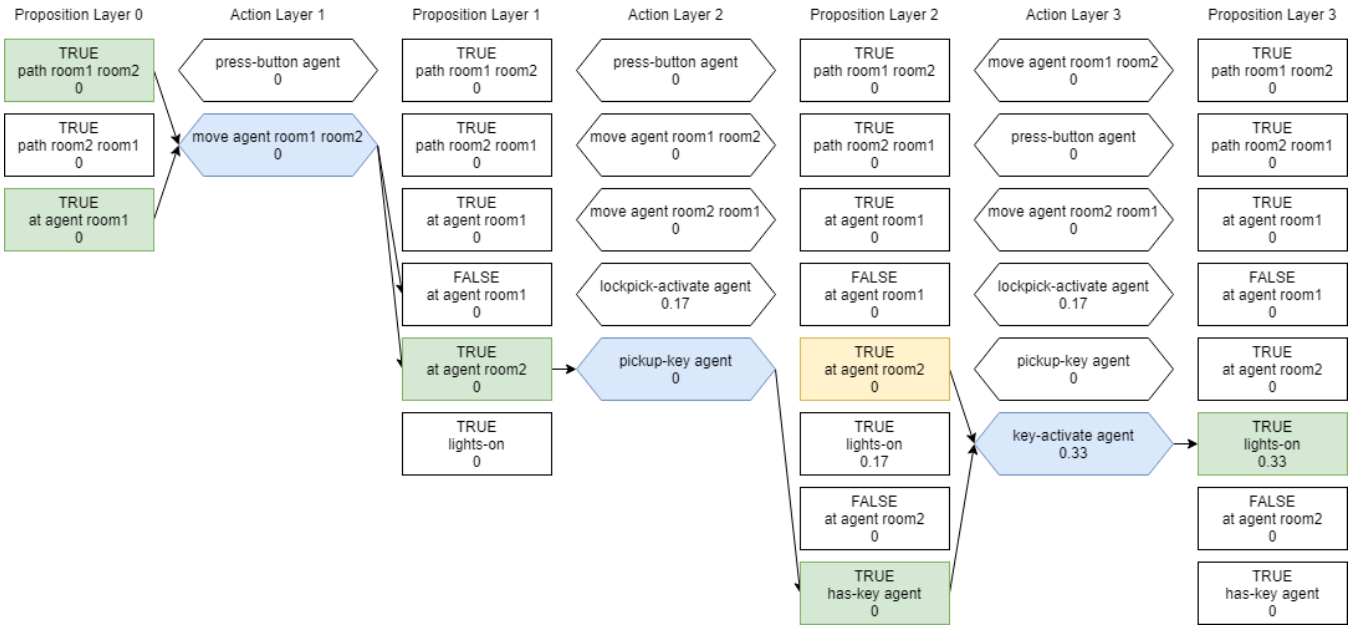

Figure 1: The relaxed plan graph and relaxed plan extraction illustration for evaluating the step of moving to room 1 in the simple "Lights" domain example.

turns 2 actions. When evaluating the move to room 1, however, RPGPREF will actually return 3 actions, as it will give a relaxed plan where the agent moves to room 2 and performs path 2. The associated RPG for this evaluation is shown in Figure 1. In this figure, the playstyle has the value of 1 for the `key-activate` action, but the evaluation would be similar regardless of value chosen as long as the value was higher in path 2. We also introduce an action called `lockpick-activate` that immediately turns on the lights in room 2 without the agent needing the key but only assign that a 0.5 playstyle value to illustrate the way the heuristic extracts a longer but higher-value plan.

This simple relaxed planning graph displays a few of the properties of this playstyle-aware algorithm. First, the goal (`lights-on`) appears in proposition layer 1. In a typical RPG heuristic, the relaxed plan graph would not be built any further than this point, as the goal can be reached by a relaxed plan. Instead, our algorithm continues to build the RPG to the fixed point which is reached in proposition layer 3.

Second, the blue steps and arrows in the graph correspond to the actions and effects selected by the relaxed plan extraction algorithm. By proposition layer 3, the goal (`lights-on`) is provided as an effect of three possible actions: `press-button agent`, `lockpick-activate agent`, and `key-activate agent`. The algorithm always selects the highest-value action that provides the goal, so `key-activate agent` is selected for the relaxed plan.

The third component that is illustrated in the above diagram is an instance of a skipped proposition on a layer, shown in yellow. In proposition layer 2, the subgoal `at agent room2` exists, but it is ignored because it exists in proposition layer 1 at equivalent or higher playstyle value. Because of this, the `at agent room2` subgoal is ignored in proposition layer 2 and is instead met in proposition layer 1. On the other hand, `lights-on` exists in proposition layer 3, 2, and 1, but because its value decreases in proposition layer 2, the algorithm chooses the goal in proposition layer 3. This process ensures that the extracted plan matches the highest playstyle value possible while still ensuring all selected steps make progress toward the goal.

While the above example is clearly very simple, it nonetheless shows some of the intricacies of RPGPREF. When running in larger domains which produce RPGs far too large to present visually, the heuristic chooses among all of the available options in a way that attempts to maximize preference values.

Empirical Evaluation

We evaluated this work by experimentally determining whether a planning algorithm driven by RPGPREF could produce simulated playthroughs such that:

1. People would recognize specific playstyles used to create playthroughs, and
2. People would differentiate playstyles in playthroughs within a single domain

The evaluation of this work was performed through a two-part human participant study. The first part of the study involved deriving playstyle values and the second part of the study was focused on evaluating the output that the algorithm produced using the playstyle values from the first part.

Domains Two domains or worlds were used in this study: one referred to as "scifi" and the other as "western".

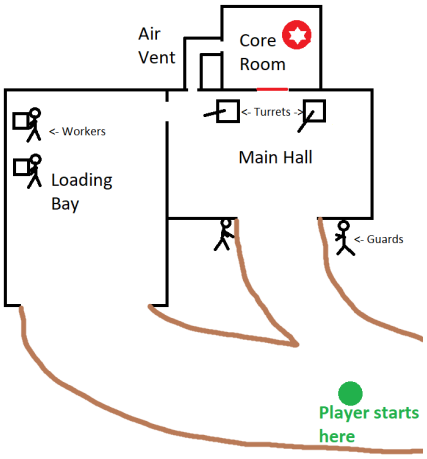

Figure 2: The scifi domain illustration.

The scifi domain was described as follows: *In this game, the player is part of the republic and must disable an empire base. In order to disable the base, the player must destroy the power core. The power core is protected by empire troops and automated guns. The base has a front gate, a loading bay, a main hall, and a core room. The front gate has guards and leads to the main hall. The loading bay has bay workers and is connected to the main hall. The main hall has automated turrets with a control panel, a locked door that leads to the core room, and an air vent to the core. The automated turrets will fire at anyone who approaches the door or the air vent. The core room has an alarm that is triggered if anyone tampers with the core that will call empire reinforcements.*

In addition, the descriptions of both domains had accompanying simple illustrations to illustrate the layout of the worlds. Figure 2 shows the illustration for the scifi domain. Both the descriptions and images were shown for both parts of the study.

Part 1: Playstyle Definition Our algorithm requires a set of proposition or action preferences, each a tuple associating an action or a proposition with a preference value for a specific playstyle. Part 1 of the study engaged participants to establish these values.

The participants were presented with the description and illustration of each domain and a list of actions available in the domain. Playstyles and their descriptions were derived from player types in Robin's Laws of Good Game Mastering (Laws 2002) and the same playstyles were used both for Part 1 of the study (deriving playstyle data) and Part 2 of the study (evaluating the output of the planner, described below). These playstyles were chosen in order to be both distinguishable and understandable by participants. The playstyles were a) *fighter* (A player who is a fighter prefers to take the direct approach, solving problems in the most direct way possible without worrying about the risks they're taking.), and b) *tactician* (A player who is a tactician prefers to take a well-thought-through approach, avoiding unnecessary risks and trying to ensure the objective is completed successfully.)

Actions were described to participants as a disjunction of all available grounded forms of an action in order to fully represent what the ungrounded version of an action involved. For example, the `shoot` action in the Scifi domain was described as *Shooting guards or turrets*. Other actions in this domain included *Shooting a door lock to destroy it*, *Triggering an alarm*, *Shooting enemy reinforcements*, *Sneaking past workers*, *Hacking automated turrets to disable them*, *Hacking an alarm to disable it*, and *Triggering a remote explosive to destroy the base*.

The participants were asked to rate how likely a player of each playstyle would be to perform a given action. This data was averaged and inserted into the domain data for the planner as preferences for the corresponding actions for each playstyle. A planner using the heuristic in this paper was run with each set of playstyle values in each domain to generate the four playthroughs for the second part of the study.

Part 2: Output Evaluation In this part of the study, participants matched the playthroughs generated using the playstyle data from Part 1 with the playstyles as they were originally described in Part 1. The participants were presented with the two playthroughs for each domain and asked to rate how well each playthrough matched the two playstyles on a six-point Likert scale. The playthroughs were converted from plan data structures to English sentences using a simple template-based approach.

Recruitment and Demographics A total of 26 participants were recruited across both parts of the study[2]. Two participants were recruited via posted flyers and 24 participants were recruited in-person. The University of Utah was chosen as the recruitment site to increase the likelihood of finding participants who were familiar with video games. The university's IRB approved of all recruitment methods and study materials.

Of the 26 participants, 5 (19.2%) identified as women and 21 (80.8%) identified as men. The majority (21, 80.8%) of participants were between the ages of 18-25 and all participants were 45 or below. Most (16, 61.5%) participants selected high school as their highest achieved education level, while 6 (23.1%) had bachelor's degrees, 2 (7.7%) had master's degrees, and 2 (7.7%) had completed a trade school program. Only 3 (11.5%) participants selected that they don't regularly play video games, while 14 (53.8%) selected that they play more than 4 hours of video games per week. 14 (53.8%) participants, when asked how familiar they were with video games, selected "I have played games across many genres", while all but 2 (7.7%) of the remaining participants selected "I have played games in a couple of genres".

[2]Six participants participated in both parts of the study. Because the first part (N=10) involved participants rating actions and the second part (N=22) involved participants rating playthroughs, we were unconcerned about this overlap. However, even with those six participants' data removed from the second part of the study, all listed results are still significant at $p < 0.001$.

Domain	Playstyle	Z	p
Scifi	Fighter	-3.96	< 0.0001
Scifi	Tactician	-3.96	< 0.0001
Western	Fighter	-3.75	< 0.0001
Western	Tactician	-3.81	< 0.0001

Table 1: Z and p values generated by Wilcoxon signed-rank tests. These tests determine the p-value for whether the playthrough generated by part 1 data for a playstyle was rated as matching that playstyle in part 2.

Domain	Playthrough	Comp	Z	p
Scifi	Fighter	$F > T$	-4.01	< 0.0001
Scifi	Tactician	$T > F$	-4.01	< 0.0001
Western	Fighter	$F > T$	-3.54	~ 0.0001
Western	Tactician	$T > F$	-3.70	~ 0.0002

Table 2: Z and p values generated by Wilcoxon signed-rank tests. These tests determine the p-value for the given comparison for the corresponding generated playstyle. Fighter is shortened to F and Tactician is shortened to T in the comparison (Comp) column.

Results

Data derived from Part 1 of the study were fed into the algorithm and it generated playthroughs that were evaluated on whether they matched the given playstyles in Part 2. The following were the playthroughs for the scifi domain:

- Generated from Fighter data: *The player shoots the empire guards at the front gate. The player enters through the gate into the main hall. The player shoots the automated turrets, destroying them. The player shoots the lock on the door to the core room, unlocking it. The player walks into the core room. The player overloads the core, setting off an alarm. The player defends the core by shooting empire reinforcements until it's about to explode. The player escapes from the base. The core explodes, destroying the base.*
- Generated from Tactician data: *The player enters through the loading bay door. The player sneaks past the bay workers to the main hall. The player disables the automated turrets by hacking their control panel. The player shoots the lock on the door to the core room, unlocking it. The player disables the core's alarm system to stop it from activating. The player places a remotely triggered explosive on the core. The player escapes from the base. The player triggers the explosive on the core, destroying the base.*

Table 1 shows how participants rated how well the playthroughs generated by the algorithm matched the given playstyle. The first stated purpose of our evaluation was to determine whether people would recognize specific playstyles used to create playthroughs, and our results show that they can. Participants overwhelmingly rated playthroughs generated with a given playstyle's data as matching that playstyle. However, Table 1 does not address our second stated purpose of the evaluation: whether people would differentiate playstyles in playthroughs within a single domain. An additional test was performed comparing the difference between participant ratings for the playstyles used in each playthrough. The results of this test can be seen in Table 2.

The results in Table 2 confirm that participants rated playthroughs generated for a given playstyle as matching that playstyle more than the ratings they gave for the other playstyle in all four cases. In other words, people successfully differentiated playstyles in playthroughs within each domain.

Discussion and Conclusion

In this paper, we presented a new heuristic for incorporating action and proposition preferences using a relaxed plan graph. We conducted a user study to verify that the algorithm could take preference data about playstyles as input and generate playthroughs that matched those playstyles as output. The results we describe above provide strong support for the claim that RPGPREF is successful in guiding a planner to generate plans which recognizably conform to and differentiate specific playstyles.

One limitation of the approach, however, is that we make a simplifying assumption that the simulated player knows all information about the domain. If a player faces two doors and one door leads them to puzzles while the other leads them toward combat, the player may not realize the difference in selecting either door, but the playstyle heuristic operates with full knowledge. In this case, a player may simply choose randomly, making this heuristic an inaccurate model of player intent. Our future work will simulate how a player plays with partial knowledge and changes their knowledge of the game world as they play.

An additional limitation involves the artificial nature of our evaluation. While the participants were able to recognize that the playthroughs matched a specific playstyle, a different approach to the evaluation could involve analyzing players in a real game environment. Participants could play a level of a game and, with data about that gameplay turned into a preference set, our system could attempt to predict what the players would do on subsequent levels. This was beyond the scope of the work on this heuristic algorithm as it would involve determining an accurate way to turn gameplay data into preference sets, but future work could involve such an approach, perhaps by leveraging data mining and machine learning techniques.

Our algorithm is a foundation for future work on preference-aware multi-agent planners (MAPs) that incorporate human players' action choice. Torreño et al. mention that "[p]reference-based MAP is an unstudied field" (Torreño et al. 2017), so we hope to be able to incorporate this heuristic in a cooperative multi-agent planner with differing playstyles defined for different agents, giving us the ability to determine how those playstyles interact while players pursue similar or identical goals. While the heuristic presented here will likely need to be a part of a planner designed for this purpose, the heuristic itself should be useful for each agent's planning process, so this is a valuable first step.

Appendix

Planning Terminology

A more comprehensive discussion of planning approaches can be found in several foundational papers (e.g., (Fikes and Nilsson 1971; Kambhampati, Knoblock, and Yang 1995; Hoffmann and Nebel 2001)). We provide a brief characterization of key aspects here:

- A *state* is typically a characterization of the truth values of relevant predicates that are taken to be true or false at a given point in time.
- An *initial state* is a special state that represents the conditions of the world at the point when an agent is expected to start taking actions leading towards its goal(s).
- A *goal state* is special state that represents a collection of conditions in the world that some agent is attempting to make true.
- A *step* is the description of an action taken in a plan leading towards the plan's goal state. Steps typically are composed of a set of *preconditions* – predicates describing conditions that must hold in the state where the step is to be executed in order for the step to succeed, and *effects* – predicates describing conditions that will hold in the state resulting from the step's successful execution.
- A *plan* is typically a pairing of an initial state, a goal state, and a sequence of steps intended to be executed from the initial state in order to change the world into the goal state.

References

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.

Bratman, M. E. 1991. Intention, Plans, and Practical Reason. *Noûs*, 25(2): 230–233.

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*.

Fikes, R. E.; and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4): 189–208.

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

Kambhampati, S.; Knoblock, C. A.; and Yang, Q. 1995. Planning as refinement search: A unified framework for evaluating design tradeoffs in partial-order planning. *Artificial Intelligence*, 76(1-2): 167–238.

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

Pizzi, D.; Cavazza, M.; Whittaker, A.; and Lugrin, J.-L. 2021. Automatic Generation of Game Level Solutions as Storyboards. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 4(1): 96–101.

Ramirez, A.; and Bulitko, V. 2015. Automated Planning and Player Modeling for Interactive Storytelling. *IEEE Transactions on Computational Intelligence and AI in Games*, 7(4): 375–386.

Riedl, M. O.; and Young, R. M. 2003. Character-Focused Narrative Generation for Execution in Virtual Worlds. In Balet, O.; Subsol, G.; and Torguet, P., eds., *Virtual Storytelling. Using Virtual Reality Technologies for Storytelling*, 47–56. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-540-40014-1.

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

Rivera-Villicana, J.; Zambetta, F.; Harland, J.; and Berry, M. 2018. Informing a BDI Player Model for an Interactive Narrative. *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play*.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587): 484–489.

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*, AAMAS '13, 603–610. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450319935.

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

Ware, S.; and Young, R. 2011. CPOCL: A Narrative Planner Supporting Conflict. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 7(1).

Ware, S.; and Young, R. M. 2014. Glaive: A State-Space Narrative Planner Supporting Intentionality and Conflict. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 10(1).

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).