Generating Dynamic Quests with Compelling Narratives Using Player Behavior

Sean Mendonca and Foaad Khosmood

Department of Computer Science and Software Engineering
California Polytechnic State University
smendonc@gmail.com, foaad@calpoly.edu

Abstract

Quests with compelling narratives are notoriously difficult to write and expensive to maintain, yet in most MMORPGs, quests must be churned out constantly to keep the players participating. Quests are often delivered in form of "instance" or parallel universes accessible only to a single player or a small party, and featuring micro-narratives that must necessarily be inconsequential to the world. Most dynamic quest generation techniques which promise to automate this process, have well known limitations such as leading to formulaic quests, being non-responsive to world events and over-relying on robotic NPCs. In this paper, we explore a different vision of dynamic quest generation, one that involves agents – including human agents – contributing dynamic and surprising elements to quests for other humans. We build a prototype multiplayer game to test aspects of this idea and report on a pilot study that demonstrates awareness of the dynamic causal chain that lead to a quest. We conduct three small user studies. While some positive results can be deciphered, the studies exposed numerous problems, and thus we consider the results incomplete. We hope this can be an early step toward realizing the goal of harnessing the dynamism of human players to generate compelling quests and narratives.

Introduction

Quests are a fundamental part of MMORPGs. Quest generation in open world MMORPGs is expensive and difficult to maintain, as they must be constantly produced to maintain the interest of players. The high cost of creating an MMORPG makes them a risky investment for developers (Bartle 2016). MMORPG Quests are often implemented as instances which necessarily exclude the vast majority of other human players in the game, and can not be consequential to the overall world narrative (if any). For example *Star Wars: The Old Republic* (Electronic Arts, 2011) uses instancing for story-related content, where player decisions lead to different outcomes in their personal story instance. However, the player decisions and story progression do not matter outside of their personal instances of the story.

Another tendency is that the single player or small party of humans who are part of the quest, must generally play the same cooperative role in the story. They are all on the same side essentially solving the same problem. There are no managed narratives that involve more than one human player in different, perhaps even adversarial roles. These issues can break suspension of disbelief, as players uncover shortcomings from ostensibly "persistent" worlds that feature multiple timelines, inconsistent experiences, and no real narrative consequence to player actions.

There have been many research and commercial attempts to use procedural content generation for story and drama related content but they usually are not able to match the quality of human-created story content (Ryan 2018). Procedural content is often relegated to side quests or minor content in commercial games such as *The Elder Scrolls V: Skyrim* (Bethesda Softworks, 2011), *Fallout 3* (Bethesda Softworks, 2008), and *Assassin's Creed Odyssey* (Ubisoft, 2018).

Many games that rely on extensive procedural content generation have been criticized as being "shallow". The same issue can seen in procedurally generated quests (Hernandez 2016). Ultimately, even PCG quests rely on a prewritten closed set of elements leading to recognizably repetitive quests and ones that are unresponsive to current world events.

The task of dynamic quest or dynamic narrative generation systems is even tougher in multiplayer RPG games. Persistent world MMORPGs, e.g. *Eve Online* (CCP Games, 2003), *Black Desert Online* (Pearl Abyss, 2015), and *Albion Online* (Sandbox Interactive, 2017) feature a world that "continues to exist and develop internally even when there are no people interacting with it" (Bartle 2004). Keeping player actions consequential in persistent worlds is much more difficult when there are multiple players acting in incoherent ways. The system must deal with the possibility that one or more of the human players won't comply with instructions or abandon the mission, potentially ruining crucial parts of the experience for others. Unanticipated agent behavior is difficult to explain in the narrative.

Our approach: user-generated content

Our approach for more dynamic automated quest narratives is to utilize user behaviors to form quests. By user behavior, we are referring to activities of in-session agents, i.e. the users generating the content are in the same game as the recipients pursuing quests based on that content. Figure 1 demonstrates the major differences between our approach

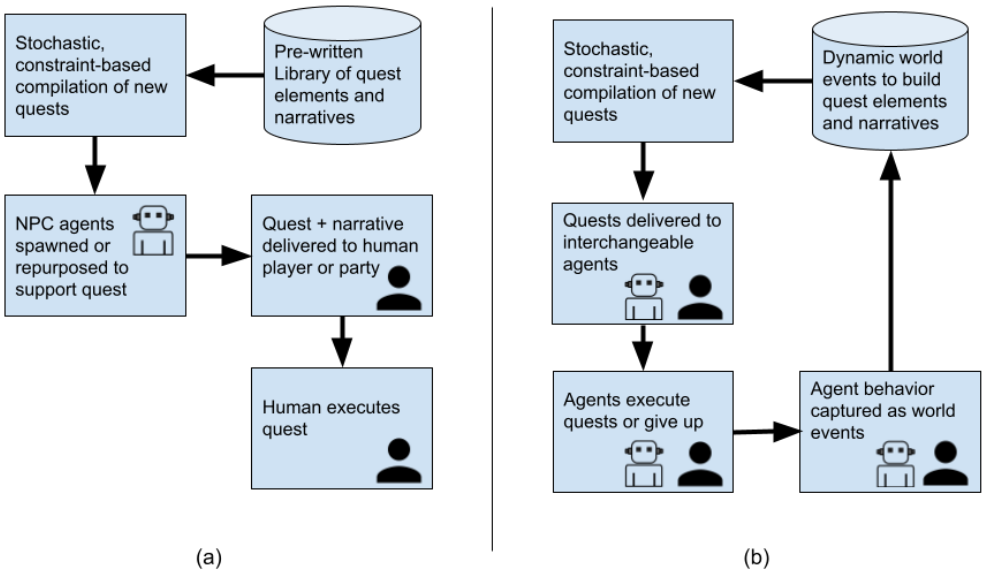

Figure 1: Typical dynamic quest generation system based on pre-written elements (a). Proposed quest generation system based on dynamic events resulting from agent/player behavior (b).

and existing approaches that rely on a static library of narrative elements.

This approach poses a number of difficult challenges which we seek to address using the experimental Panoptyk framework. The framework, which is also developed by the authors and used in a few other projects(Miller et al. 2019), is explained below in the Related Works section. Panoptyk allows us to set up experiments around just the design of quests and narratives without having to create a full scale MMORPG.

We conduct three user studies, but encounter problems that basically invalidate the results. However, we do share our experience and hope this work can contribute to better studies in the future.

Related Work

Story and narrative generation

Inspired by *Morphology of the Folktale* (Propp 2010), Joseph Grimes designed a system in early 1960s that would use the narrative structures to generate stories. His system predated other story generators but was not published or discussed in detail until Grimes' interviews with James Ryan many years later (Ryan 2017).

A few years later, *TALE-SPIN* (Meehan 1977) was built as a system to generate simple stories based on resolving a problem/goal of a main character. The program was able to use location, personality, relationships, and world state to plan out the actions each character takes. The resulting "stories" were a description of the modeled worlds and logical actions characters took to fulfill their goals. However, even though characters acted in a reasonable way to achieve their goals, there was no way for the system to differentiate a compelling story from a mundane account of facts and actions that characters took to accomplish simple goals. Nonetheless, *TALE-SPIN* is considered the first planning-based approach to story generation and is frequently referenced by scholars in the field as such (Shaker, Togelius, and Nelson 2016). Other planning based narrative generation programs that followed *TALE-SPIN* include but are not limited to *AUTHOR* (Dehn 1981), *UNIVERSE* (Lebowitz 1985), and *Minstrel* (Turner 1993).

Quest generation

The *TRUE STORY* (Pita, Magerko, and Brodie 2007) system attempts to generate compelling quests for MMORPGs by using the past quests and/or history of players. The system is able to create "kill," "steal," "discover," and "retrieve" quests based on positive or negative history with other players or items. The system also takes into account other factors such as the player's skills when assigning appropriate quest targets. Unfortunately, the authors did not discuss any form of user testing or other validation metrics. They admit the system did not have a way to determine the importance of different pieces of history and therefore struggled to generate a meaningful chain of quests.

Doran and Parberry describe a prototype quest generator based on their analysis of over 750 quests from *Eve Online* (CCP Games, 2003), *World of Warcraft* (Blizzard Entertainment, 2004), *Everquest* (Sony Online Entertainment, 1999), and *Vanguard: Saga of Heroes* (Sony Online Entertainment, 2007). The authors categorize quests from said games into nine different categories representing the motivation behind the quests (Doran and Parberry 2011). The motivations were then associated with tasks that they call "strategies" that could be fulfilled with a sequence of actions. Their final prototype is able to randomly generate quests from their

database of motivations and strategies as well as list out actions that players would need to do to complete them. In a followup paper, the authors describe the implementation of their quest generator into the MMORPG *Everquest* (Doran and Parberry 2015). However, their integration still requires a human designer to supply characters and dialog to be used by the quests. The generated quests also required a very specific chain of actions to be fulfilled due to the use of partial-order planning.

Dwarf Fortress (Bay 12 Games, 2006) is often seen as one of the most successful and influential practical applications of emergent narrative and procedural content generation (Ryan 2018). Hundreds of years worth of actions, including the rise and fall of civilizations, is generated through simulation. Rather than trying to explicitly generate stories, the game relies on the strength of the simulation to naturally create interesting events and situations that can be retold as a story. Procedurally generated quests appear in adventure mode, a special mode where the player controls a single character rather than an entire fortress. Quests in this mode are given by an omniscient commander NPCs that create quests based on events affecting their faction (Dwarf Fortress Wiki).

MKULTRA

MKULTRA (possibly named after but is unrelated to the CIA mind control project of the same name) is an experimental system by Ian Horswill designed to explore several AI-based game mechanics (Horswill 2015). Game mechanics are designed around interacting with NPCs that have generative reasoning and natural language capabilities. The player controls their character by specifying goals and actions through typed natural language. The game is a "mystery-and-detection" game where the player needs to solve secrets. Players are able to influence the behavior of NPCs by injecting false beliefs directly into the NPC's knowledge base. Characters in the game only complete requests that do not conflict with any of their beliefs. The game was intended to serve as a open-source platform for AI and interactive narrative researchers.

In a follow-up paper, Horswill addresses the successes and failures of MKULTRA (Horswill 2018). The Unity Prolog interpreter built for the project ended up being utilized for several projects, including the commercial game *Project Highrise*. The performance of a definite clause grammar (DCG) parser turned out to be sufficient according to Horswill. Furthermore, NPCs characters using reactive planning were able to excel at accomplishing simple tasks typed out by players. Horswill gives the example that typing "can I have an apple?" would result in the NPC going to the kitchen, opening the refrigerator, taking an apple, walking back to the player, and finally, giving it to the player. Unfortunately, many players struggled to correctly interact with the system. Players would often type incomplete commands and/or explore the environment without interacting with NPCs. They also struggled to determine what commands were valid and wasted time attempting different sentences. The limited scope of NPC knowledge bases was also a major problem; NPCs were unable to appropriately respond to prompts or behaviors outside the scope of their limited understanding of the world. Horswill concludes that attempting to portray NPCs as lifelike characters directly conflicted with the limited ways in which the game could actually be solved. This ultimately led to player confusion and inability to solve the puzzle.

Panoptyk framework

Panoptyk is an open-source experimental framework meant to create MMORPGs with persistent worlds, where NPCs are designed to be indistinguishable from human players. NPCs are given the same information and capabilities as human players, and the game is heavily based around the creation, possession, and exchange of information around the game world. All "agents" (human or bot/NPC) use the same interface to connect to a main game server and perform activities, such as movement, exchanging information or inventory items. The server does not distinguish between human or bot. A separate HTML-based client is used for actual humans but it is merely a layer for the same API calls the bots use. Bots are meant to be launched by participants with code executing on remote machines.

Players, once joined, form or join factions that have conflicting overall goals, and must engage in navigation, exchange, intrigue and deception to fulfill their faction's objectives. The system supports a map in form of a set of connected spaces. It allows agents to enter and exit spaces and observe the actions of other agents in the same space.

Experimental Setup

We create a scenario to test out the quest generation system within the Panoptyk framework. We create the fictional town Bentham (Figure 2) and two factions of agents: Town Guard and Thieves Guild. The world is populated with at least 10 AI-controlled NPCs, though the human players were not told of this.

Faction setup

In our prototype, a pre-programmed NPC faction leader agents assign appropriate quests to their faction members in return for rank advancement and gold. These tasks align with their faction's goals and are reactive towards current events in the game. Since quests are designed around the agent-generated information, quests can be dynamically generated as long as new information is being created by other agents. We theorize that using players' actions to determine quests will increase their personal connection to quests and reduce the repetitiveness often noticed in other MMO quest systems.

Town guard faction The town guard faction leader constantly assigns quests to combat illegal activity around the city. First it assigns quests to arrest agents that have committed crimes. After assigning quests to apprehend all active criminals the town guard leader will assign quests to collect known contraband and deliver it for safekeeping. Finally, if no crimes or illegal items are detected, it will assign a quest for town guard agents to report any illegal actions or items.

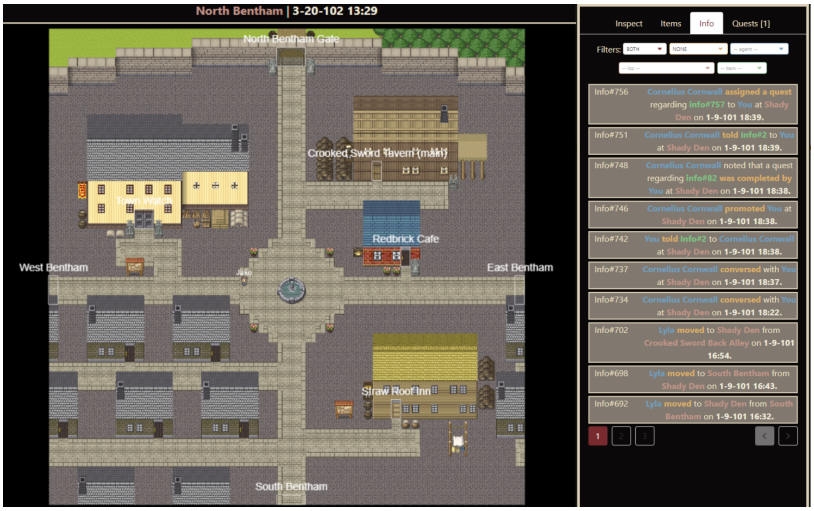

Figure 2: Web-based client view of the fictional town of Bentham (left). Client information panel detailing some recent information units (right).

Four "informant" agents are created to supply the guard faction leader with information and receive quests.

Thieves guild faction The thieves guild leader is primarily concerned with the collection of illegal goods for itself. The first priority of thieves guild leader is to assign quests to "take revenge" against members of the town guard who have recently arrested members of the thieves guild. The "revenge" quest currently just means identifying and attacking the responsible guard. Once all possible "revenge" quests have been assigned the thieves guild leader focuses on assigning quests to loot the most valuable items that it knows about. Occasionally, it may decide to assign a quest to reward an agent that have been helpful to the faction. Finally, if it cannot create any other valid quest from its current knowledge it assigns a quest to search for new treasures to acquire. Four NPC faction members are also available to receive quests the same way player faction members do.

When possible, both faction leaders prioritize assigning quests that are based on information directly relevant to the agent they are talking to. For example, revenge quests for the thieves factions are always assigned to the agent that has been arrested by the targeted member of the town guard. The same logic is used when assigning item retrieval quests; the questing agent is informed it is being asked to retrieve the item because it is the one that told about the whereabouts of that item. This rule is not absolute though, factions leaders will assign these followup quests to other faction members if it has no other available quests and the preferred agent is busy with another quest or otherwise unavailable. Ideally this system is intended to give players the impression that quests are based on their personal experiences.

Quest Generation System

Quests are represented in special information units that signify a goal to complete. They are generated based on the current information knowledge of the quest giver, the quest giver's opinion of other agents in the world, and their unique pre-defined goals. The quest giver's information knowledge is based entirely on events that occur in the game. Since quests are designed around the agent-generated information, quests can be dynamically generated as long as new information is being created by players. Quest assignment logic for quest givers is shown in Algorithm 1 and Algorithm 2. In total there are 7 different types of quests that can be generated by the faction leaders based on their current logic. This may seem like a small number but each quest type can require a wide variety of different player actions to complete and lead to different consequences. For example, the quest to retrieve an item can result in a player searching for it, stealing it, buying it, picking it up, or simply handing it over. Quests can lead to different follow-up consequences depending on their targets and how a player chooses to complete them. The information system of Panoptyk allows for easy creation of new quests types once gameplay actions are added to support them.

Information representation

In general, information representation in Panoptyk is based on first-order logic predicates. This representation is designed to make it as easy as possible for NPC agents to reason about the data they receive. All events that occur in the world generate an information object. These objects link the action performed with the relevant variables (agent(s), location, time, item(s), etc.). Every possible action (shown in table 1) is codified and new actions specific to other games can be supported easily.

The server keeps a master copy of every piece of generated information, all information distributed to clients are a reference to the master server copy. Panoptyk supports the the usage of partial information, information where one or more fields are masked to its owner. Masked information still references the original server master copy, but the masked fields are not sent to the client. The information sys-

Algorithm 1: Thieves Guild quest assignment logic

Require: target agent to give quest
Ensure: valid quest to give to target agent
  if an opposing agent has done an action to harm target agent **then**
    return a revenge quest targeting the opposing agent that refers to the harmful action done to target agent
  else if some other agent's action(s) have pleased the faction leader **then**
    return a gift delivery quest targeting the other agent that refers to how they pleased the faction leader
  else if there is a valuable item that we don't own **then**
    assign item retrieval quest that refers to last known information about that item
  else
    assign generic quest to discover new items in the world
  end if

Algorithm 2: Bentham Guard quest assignment logic

Require: target agent to give quest
Ensure: valid quest to give to target agent
  if there is a crime that has gone unpunished **then**
    return an arrest quest targeting the criminal and referring to the crime committed
  else if there is an illegal item that is not currently impounded by the Town Guard **then**
    return an item retrieval quest that refers to last known information about that item
  else
    return a quest to patrol the town and report any illegal actions
  end if

tem gives AI bot clients sufficient details to function without the server's intervention, thus maintaining our decentralized system that allows external agents to shape the game world.

User study

Three user studies were carried out to validate the quality of generated quests. The questions and format of the study changed slightly between each iteration to ameliorate issues as they appeared. In all studies, each subject was placed on either the Bentham City Guard or Thieves Guild factions. Subjects were expected to play through a few quests and then answer a survey on their experience. All Versions of the survey consisted of five different sections to evaluate multiple parts of Panoptyk and the scenario. Free-response questions are grouped into categories of expected answers. Users spent 30-60 minutes in each study.

First study The first study was offered to a college class of Interactive Entertainment Engineering students. The testing environment was chaotic due to multiple groups of students trying to get their games tested. Subjects were given basic instructions to follow the quests issued by their assigned faction and attempt to complete them to the best of their ability. Unfortunately there were several technical and design issues that hampered the testing session. This was our first attempt to utilize more than three human testers in a single scenario, and as a result the number of AI agents and scenario size were poorly balanced for the increased player activity. The user interface of Panoptyk at this point was very rudimentary and lacked an in-game tutorial. These issues meant almost all users struggled to play the scenario.

Table 1: List of actions and predicates: (T)ime, (A)gent, (I)tem, (L)ocation, (Q)uantity, I(N)formation

Action	*Predicate Variables*
Move	(T,A,L,L)
Pickup	(T,A,I,L,Q)
Drop	(T,A,I,L,Q)
Steal	(T,A,A,I,L,Q)
Pay	(T,A,A,L,Q)
Arrest	(T,A,A,L,N)
Assault	(T,A,A,L,N)
Converse	(T,A,A,L)
Gave	(T,A,A,I,L,Q)
Ask	(T,A,A,L,N)
Tell	(T,A,A,L,N)
Assign Quest	(T,A,A,L,N)
Fail Quest	(T,A,A,L,N)
Complete Quest	(T,A,A,L,N)
Show Possession	(T,A,I,L,Q)

Second study The second study was completed over voice chat with five members of a college game development club. The scenario and clients interface were significantly refined for this version. These changes included more detailed quest reasoning, many bug fixes, and a new "Help" window designed to teach the UI and give tips to confused players. Despite many hours of bug fixing and informal testing with other people, this session still suffered from some issues. There were two servers crashes caused by one or more testers not correctly following given instructions. Fortunately, once the problem was found the rest of the test was able to proceed.

Third study The final session was designed to address the majority of the issues from the previous sessions. Six volunteers conveyed their interest in participating. Testers were assigned to two separate online sessions in groups of three in order to minimize the chaos that was present in other tests. It was also decided that the help interface was enough to quickly educate players on how to operate the UI, so a guided tutorial over screen-share was made part of the testing process. Instructions were emailed to subjects multiple days before their assigned session. Frustratingly, only 1 out of 6 volunteers actually showed up to their assigned session. While the participation rate was disappointingly low, the lone tester did not run into any UI or technical glitches during his testing session.

Evaluation

We ran multiple internal tests to make sure that the questing system was working as intended and that the AI faction

members were acting in a manner befitting their current situation and faction personality. Majority of our early testing was focused on finding ways to have new information flow to faction leaders. This was essential because unlike most games, our quests have to be created entirely from the faction leader's internal knowledge model.

Validation on quest generation with limited information

To show that quest generation is possible with limited information, quest givers must always be able to assign a quest. To prove this, we start a fresh server instance in which no agent has any information given by default. The quest giver is able to, without knowledge of anything other than the room it is in, request a conversation with any valid agent in its room and assign a generic quest to gather more information. This generic quest is possible by creating a type of information predicate without specifying its action. For example, a quest command of TILQ could be completed with any information that has a time, item, location, and quantity. Both AI agents and human clients are able to interpret this as a quest to find any information that has the given properties. Therefore quests are able to be generated with only basic knowledge of the structure of information.

Validating that relevant quests are generated

We define relevant as "based on events happening in the game and related to the overall goals of the quest giver." In order for a quest to be based on events happening in the game, the quest must have been created from an action that occurred during gameplay. From the generic information gathering quest, quest givers are able to gain information about events occurring in the world. Quest givers use the logic defined in either Algorithm 2 or Algorithm 1 to create a valid quest that aligns with their goals. The generic information gathering quests themselves do not meet our definition of "relevant" but allow for quests that do to be generated. Internal testing found that the full range quests defined in said algorithms were able to be assigned and completed. We can conclude that, given our definition of "relevant," the system is able to produce relevant quests. This was confirmed in the internal validation test, data represented in Figure 3 shows that the majority of the assigned quests referenced an in-game event.

Validation that quests have world consequences

For the purpose of internal validation, we consider quest related "consequences" as in-game quests assigned because of any actions done as part of a previous quest, thus building a causal narrative. We can show that quests cause consequences that can affect both the questing agent and other agents in the game. It is straightforward to see how agents experience personal consequences from completing quests, the generic information gathering quest often has the consequence of a followup quest referencing the information turned in as part of the information gathering quest. In specific cases, such as when an agent does an illegal action to

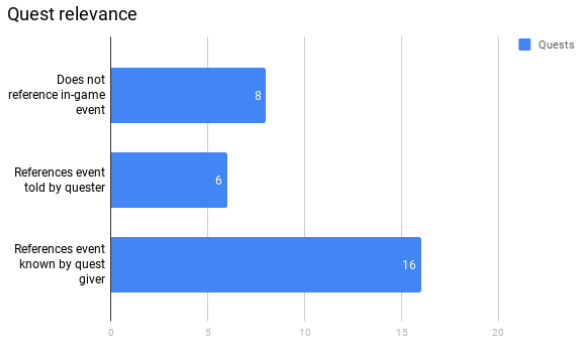

Figure 3: Internal test on quest relevance with a scenario using 10 AI agents and 2 humans

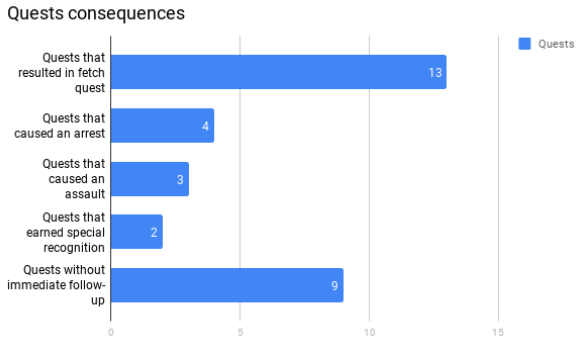

Figure 4: Internal test on follow-up consequences to quests generated with a scenario using 10 AI agents and 2 humans

complete a quest, another agent can experience the consequence of having to hunt down and arrest the first questing agent. This in turn can cause additional consequences for the first quest agent, where they will be directed to take revenge on the agent that arrested them (another illegal action). There is also the positive consequence of a gift quest that can be assigned to reward an agent that has done things to please a quest giver. Internal testing shows that all of the described quest consequences regularly occur, proving that quests in the system have what we have defined as "consequences". To show that quests cause follow-up consequences, we tracked consequences on the 30 quests generated in the internal test. Figure 4 shows the resulting follow-up quests from the internal test; a follow-up quest was any quest that referenced an event turned in to complete a previous quest.

Results

Results of first study

Issues with the UI meant that most users were not able to provide much feedback on quests. Only 2 of the 7 testers claimed that playing was an enjoyable experience. 6 out of 7 testers claimed that the UI was a weakness of the game. In a more positive note, 5 out of 7 testers claimed that the

information system was a strength of the game, which is important as the information system is core to the Panoptyk engine. In terms of quests, only 3 of the 7 testers were able to complete any quests; only one of them was able to complete more than one quest. Unsurprisingly, 6 out of 7 testers to describe quests as "a little hard" or "way too hard" and resulted in mostly neutral or negative responses for most questions related to quests. Fortunately, the player who was able to complete the most quests had extremely positive feedback for the quests. Feedback on the faction section showed that only 1 of the 7 testers thought that assigned quests did not make sense for their faction identity. 5 of the 7 testers felt that their actions had an impact on other players in the game, multiple players noted that other players could pick up items required for their quest. Three players felt that there were not enough actions in the game, while another three felt that the actions were too complicated or hard to use.

Results of second and third studies

The results of the second and third studies are combined since there were few participants and only minor feature changes between the two sessions. No aspect of the game was universally considered a strength or weakness. 2 of the 6 players were able to deduce that the quests were based on events happening on the game. Every participant was able to complete quests, with 4 out of 6 of them completing 5 or more quests. One player turned in 55 quests due to an exploit that allowed him to turn in the same quest repeatedly. Of the six participants, three of them found quests "a little hard," one found quests "way too hard" and one found them "a little easy." Every participant agreed that quests were based on events happening in the game. Unfortunately, due to the limited possible action space for quests, all but one user found quests repetitive. Participants were divided on whether quests had meaningful objectives or not. Two of them thought they were not meaningful, three were neutral, and one thought they were meaningful. Participants were polarized on whether they received any positive or negative consequences from quests, 2 agreed or strongly agreed they did and 2 strongly disagreed they did. In the free response section asking for participants to describe interesting encounters they had as a part or result of a quest, one participant mentioned he got stuck getting assigned the same quest for the whole game, another participant noted they did a lot of "thanking" quests, and the remaining participants talked about an arrest event. In the feedback section, there was four complaints related to bugs or UI issues. The remaining participant wanted more detail in the empty areas of the city.

Only one participant was able to complete the extra questions designed for the third study. The participant agreed having quests based on player input had a positive impact on his experience. When asked about the advantages of such quests he wrote "It helps make the story unique ... It can also help the player get more involved in the story if they are having a direct impact on the direction that it is going, instead of a quest-on-rails storyline." which was the exact intention behind the system. When asked about the disadvantages of the system he responded "I went and looked at a lot of different items before coming back to the faction leader. Since I had

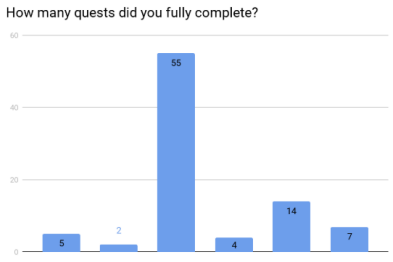

Figure 5: Second study quests completed

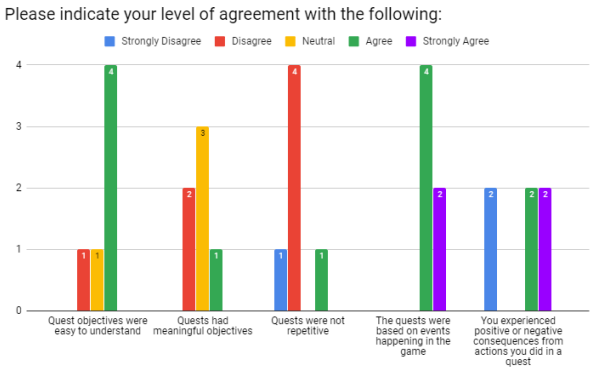

Figure 6: Second study quest questions

done that, I had a lot of options to choose from for turning in the first quest.

The feedback towards factions was similar to the first study. No participants thought that the quests assigned were not appropriate for their faction. For additional feedback, one participant complained that there was not enough backstory given. Another participant never ended up interacting with someone from a different faction, this player was also the player assigned an abnormal number of "thanking" quests. The feedback on impact towards other players correlated heavily with the question that asked about quest impact. The only participants who disagreed that their actions impacted other characters in the game were the players who were not assigned quests targeting other players. When asking for additional feedback on gameplay actions, two participants said there were enough actions to make gameplay engaging. Three participants claimed that they had issues with either the information system or UI.

Analysis

Overall, the results from user testing show positive trends with room for improvement. The majority of users who did not experience crippling bugs were able to perceive that the generated quests were relevant to the events they were experiencing, and that their actions could have consequences that affected them later. The action/consequence implications of quests shows that our system contains the foundations needed to generate immersive quests. All players who experienced arrest-related quests talked about arrest-related encounters when asked about interesting encounters related

to quests. This shows that players associated events around the central plot point of being arrested.

The mixed feedback regarding the meaningfulness of quest objectives was likely due to a number of factors. A few players ran into critical bugs that prevented them from completing quests. Players who were able to experience quests were not guaranteed to experience all possible types of quests due to the dynamic nature of their generation. In the second study, a few of the players were able to gather all the items around the world and deprive the slower players of any chance to complete a quest. This issue was corrected by restrictions to carry capacity in the third study, but was not able to be thoroughly tested due to the lack of participants. Another factor that may have hampered the meaningfulness of quests objectives was the lack of backstory and context surrounding the scenario.

Complaints about the repetitiveness of quests are unavoidable given that limited gameplay is possible in this early build of the game. The main gameplay loop of finding items did not really have any exciting moments associated with it. The fact that no player commented on the different rewards they received from completing quests probably means they were uninteresting enough to go unnoticed. The way quests objectives were displayed could have been another reason for why quests were perceived as repetitive; quest objectives always displayed the end goal rather than steps that a player needed to take to complete it. For example, the item quests always required a player to given an item to their faction leader, but did not display any intermediary objectives leading up to the final objective. Assigning intermediary objectives may have helped players notice that the actions they were taking to complete quests were unique and dependent on the overall state of the world.

Conclusion and Future Work

We have outlined a decentralized dynamic quest generation system based on the observations and inputs of agents in MMORPGs. This system was able to dynamically generate personalized quests based on the experiences of the agents interacting with it. We show, through a user study, that humans recognized that our dynamic quests were based off of in-game events rather than procedurally generated facts. We also show that players noticed positive or negative consequences from actions they did as part of a quest. The consequences of quest actions show the building blocks of generated drama and story. The drama of "arrest" quests led to many players describing their encounters in the context of that central plot point.

For future work we intend to overhaul the information management interface. By design there are thousands of information units available to the human user, and the vast majority are useless, yet must be accessible from the interface. Much more player testing is needed as we only had small studies so far. The natural language descriptions of game information units also must improve in order to organically and predictably describe the information to the user. In addition, the relatively basic gameplay offered by the current engine made it difficult to offer varied quests that humans player consider interesting. Gameplay mechanics based on traditional RPGs mechanics, such as unique abilities that can be leveled up, basic combat between adversaries, and stealth actions would likely make the game more interesting to the average gamer. We hope additional gameplay in Panoptyk will open up more opportunities to create dramatic plot points.

References

Bartle, R. A. 2004. *Designing virtual worlds*. New Riders.

Bartle, R. A. 2016. *The Decline of MMOs*, 303–316. Cham: Springer International Publishing. ISBN 978-3-319-40760-9.

Dehn, N. 1981. Story Generation After TALE-SPIN. In *IJCAI*, volume 81, 16–18.

Doran, J.; and Parberry, I. 2011. A prototype quest generator based on a structural analysis of quests from four MMORPGs. In *Proceedings of the 2nd international workshop on procedural content generation in games*, 1–8.

Doran, J.; and Parberry, I. 2015. A server-side framework for the execution of procedurally generated quests in an MMORPG. In *GAMEON'15-Proceedings of the 16th Annual European Conference on Simulation and AI in Computer Games*, 103–110.

Dwarf Fortress Wiki. 2014. DF2014:Quest.

Hernandez, P. 2016. The Internet Loves Making Fun Of Fallout 4's Preston Garvey. *Kotaku Australia*.

Horswill, I. D. 2015. MKULTRA. In *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference*.

Horswill, I. D. 2018. Postmortem: MKULTRA, An Experimental AI-Based Game. In *Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Lebowitz, M. 1985. Story-telling as planning and learning. *Poetics*, 14(6): 483–502.

Meehan, J. R. 1977. TALE-SPIN, An Interactive Program that Writes Stories. In *Ijcai*, volume 77, 91–98.

Miller, M.; Mendonca, S.; Philliber, N.; and Khosmood, F. 2019. Panoptyk: information driven MMO engine. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, 1–4.

Pita, J.; Magerko, B.; and Brodie, S. 2007. True story: dynamically generated, contextually linked quests in persistent systems. In *Proceedings of the 2007 conference on Future Play*, 145–151.

Propp, V. 2010. *Morphology of the Folktale*, volume 9. University of Texas Press.

Ryan, J. 2017. Grimes' fairy tales: a 1960s story generator. In *International Conference on Interactive Digital Storytelling*, 89–103. Springer.

Ryan, J. 2018. *Curating simulated storyworlds*. Ph.D. thesis, UC Santa Cruz.

Shaker, N.; Togelius, J.; and Nelson, M. J. 2016. *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*. Springer.

Turner, S. R. 1993. Minstrel: a computer model of creativity and storytelling.