MEASURING COOPERATIVE BEHAVIOR IN CONTEMPORARY MULTIPLAYER GAMES

Martin Ashton

Master of Science

School of Computer Science

McGill University

Montreal, Quebec

2012-08-12

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

Copyright \bigodot 2012 Martin Ashton

DEDICATION

This work is dedicated to my friends Pete, Dave, Kevin, Vanessa, Steve, Mike, Nadina, John, Chris, Max, Dan, Francine, Peter, Daniel, Michael, Dahlia, Sabrina, Michaela, Katrina, Diana, Stevie, Kathy, Bronson, Jess, Theo, Cynthia, Ben, Shayne, Paul, Andre, Fred, Justin, J.P., Aly, Carl, Udhay, Phil, Matt, Simon, Julien, and the rest of the Koalas.

ACKNOWLEDGEMENTS

This work was supported, in part, by the Natural Sciences and Engineering Council of Canada (NSERC), and Le Fonds de Recherche du Québec - Nature et Technologies (FQRNT). My particular thanks are extended to Prof. Clark Verbrugge for giving me the opportunity to research the exciting domain of modern video games, and to my parents for teaching me the importance of continuously exceeding my own limits. Lastly, I would like to thank the developers at Bungie, Blizzard, and Valve for making such excellent games.

ABSTRACT

Social aspects of multiplayer games are well known as contributors to game success, with online friendships and socialization expected to expand and strengthen a player-base. Understanding the nature of social behavior and determining the impact of cooperation on gameplay is thus important to game design. In this work, we make use of data exposed through in-game and web-based API's of two contemporary multiplayer games, WORLD OF WARCRAFT and HALO: REACH. We use this data to investigate the extent of cooperation among players and the effect on individual player behavior. We moreover show how the quantitative assessment of cooperative behavior can be used to isolate potential problem areas in games which may require additional balancing.

We first monitor group health and position to measure the *pacing* of a cooperative scenario in WORLD OF WARCRAFT. We measure a scenario's pacing as the temporal progression of its difficulty, which directly reflects the required level of cohesion and coordination among the players in a group. Our results verify the informal perception that statically designed content becomes increasingly trivial as players obtain stronger stats, thus reducing the need for cohesion. Direct quantification of this behavior, as enabled by designs such as ours, allows for online, adaptive pacing that should better foster player community by consistently emphasizing the need for communication.

The benefits of actual group behavior also has a reverse impact on game design. In our experiment involving HALO: REACH, our results demonstrate that players who enter as a group into the multiplayer matchmaking system have, on average, a significantly higher win-to-loss ratio than players who enter the matchmaking system alone. This gives them an advantage over less social players, and thus attests to the potential for refinement in group matchmaking techniques. In addition, our exploratory principal component analysis of individual player performances reveals a set of novel player types adapted to the multiplayer context and quite distinct from player types found in other game genres.

From a general standpoint, the data collection techniques outlined in this thesis reveal the use of publically-accessible game APIs as a relatively unexplored yet promising source of insight into real-world gameplay behavior. Our results serve as evidence for two widely-assumed notions of multiplayer game design; the first, that static game content adversely affects a game's replayability and ultimately lessens the need for communication and cohesion among players. The second, that coordination among players provides a significant advantage over those who choose to play independently in a team-based setting.

ABRÉGÉ

Les interactions sociales entre les utilisateurs de jeux vidéo multijoueurs contemporains contribuent largement à la propagation et à la longévité de ces derniers. La compréhension des facteurs qui se lient à la promotion d'interactions sociales au sein de ces environnements est donc importante à leur développement. Dans cette thèse, nous recueillons des données à partir d'interfaces de programmation de deux jeux multijoueurs contemporains: WORLD OF WARCRAFT et HALO: REACH. Nous analysons ces données afin d'évaluer l'effet global du comportement coopératif, ainsi que son effet sur le comportement d'individus. De plus, nous démontrons que la mesure quantitative de comportements coopératifs peut aider à l'identification de fautes systémiques d'un jeux.

En premier lieu, nous mesurons les points de vie et la position des membres d'un groupe de joueurs pour évaluer le débit d'un scénario coopératif de WORLD OF WARCRAFT. Nous définissons ce débit en fonction de la difficulté du scénario par rapport au temps. L'achèvement d'un scénario à débit intense impose ainsi un niveau de communication plus élevé entre les membres du groupe. Nos résultats appuient d'ailleurs la perception informelle que les jeux conçus avec des environnements et des ennemis non-adaptifs perdent l'intérêt des joueurs lorsque ceux-ci deviennent trop puissants. De plus, cet accroissement en puissance réduit le nombre d'interactions sociales en diminuant l'exigence de la communication entre les joueurs. En deuxième lieu, nous observons les conséquences de la coopération entre les joueurs de HALO: REACH. Les données recueillies dans ce contexte suggèrent que les joueurs qui entrent en groupe d'amis dans le système d'établissement de parties ont de plus fortes chances d'obtenir une victoire que ceux qui s'y introduisent individuellement. Nous découvrons ainsi une faute potentielle de ce système d'établissement de parties qui favorise les joueurs plus sociaux au détriment des joueurs plus solitaires. De plus, nous appliquons une analyse des composantes principales (PCA) sur les résultats moyens de chaque joueur, ce qui révèle un ensemble de descripteurs adaptés au contexte multijoueur, très distinct des descripteurs attribués aux joueurs d'autres types de jeux.

D'un point de vue global, quoique les interfaces de programmation de jeux soient relativement inexplorées, notre méthodologie démontre que celles-ci offrent une panoplie d'informations liées aux comportement de joueurs. Nos résultats supportent d'autant plus deux notions informelles enracinées dans le design de jeux multijoueurs – la première dicte que les environnements statiques agissent contre la rejouabilité d'un jeu, et que ceux-ci réduisent ultimement les besoins de communication et de cohésion entre joueurs. Dans un contexte d'affrontements d'équipes, la deuxième notion soutenue par nos données suggère que les joueurs coordonnés en groupe ont un avantage inné par rapport aux joueurs plutôt indépendants.

TABLE OF CONTENTS

DED	ICATI	ON	ii		
ACKNOWLEDGEMENTS ii					
ABS	ABSTRACT				
ABRÉGÉ					
LIST	OF T	ABLES	х		
LIST	OF F	IGURES	xi		
1	Introd	uction	1		
	1.1 1.2	Contributions	4 4		
2	Measu	ring Cooperative Behavior in World of Warcraft	5		
	2.12.22.32.4	Background2.1.1World of Warcraft2.1.2WoW AddOnsMethod2.2.1Orunj AddOn2.2.2Measuring Player Intensity2.2.3Minimum Enclosing Circle ProgressionExperimental Results2.3.1Experimental Context2.3.2Intensity Results2.3.3Pacing Results2.3.4Discussion	6 9 10 13 14 15 17 22 26 30		
3	Measu	ring Cooperative Behavior in Halo: Reach	31		
	3.1	Background	32		

		3.1.1 Halo: Reach
		3.1.2 Stats API 35
	3.2	Data Collection
	3.3	Data Analysis
		3.3.1 Player Graph
		3.3.2 PCA
	3.4	Discussion
4	Relate	d Work
	4.1	Metric-Assisted Game Development and Adaptive Gameplay 64
	4.2	Measuring Player Behavior and Player Types
5	Conclu	sions and Future Work
	5.1	Conclusions
	5.2	Future Work
Refe	rences	

LIST OF TABLES

Table		page
2-1	MEC Metrics and Intensity	28
3-1	Rounds of data collection	40
3-2	Node Degree Distributions	43
3–3	Round 1: simplified principal components	59
3-4	Round 2: simplified principal components	60
3–5	Round 3: simplified principal components	61
3–6	Round 4: simplified principal components	62
3–7	Simplified principal components from the merged data set	63

LIST OF FIGURES

Figure		page
2-1	Orunj system architecture	11
2-2	WoW positional data visualization using the Lua-based LOVE frame- work.	12
2-3	Playable area of the first floor of Utgarde Keep	19
2-4	Player positional data from the Violet Hold dungeon	21
2-5	Player position and MEC plot of a sample Utgarde Keep run. $\ . \ . \ .$	23
2-6	Player intensity plots	25
2-7	Minimum Enclosing Circle (MEC) heatmaps	27
3-1	Halo: Reach multiplayer data analysis system	36
3-2	A conceptual simplification of the sampled player graph	41
3–3	Node degree distribution of Rounds 1 through 4 of data collection	42
3-4	Edge weight distributions in Round 1 of the data collection	45
3-5	Edge weight distributions in Round 2 of the data collection	46
3-6	Edge weight distributions in Round 3 of the data collection	47
3-7	Edge weight distributions in Round 4 of the data collection	48
3-8	Average component size versus pruned edge weight, Rounds 1 and 2 $\ .$	50
3–9	Average win ratio versus number of friends in a game	51
3-10	Summed PC weights versus group win ratio (top 10)	56
3-11	Summed PC weights versus group win ratio (bottom 10)	57

CHAPTER 1 Introduction

Multiplayer games benefit from strong social engagement; a large and cohesive player community is essential for providing players with an abundance of other players to interact with as team-mates or opponents, and is also seen as mechanism to encourage player retention. Such player bases can be cultivated by the necessity to cooperate and communicate in games, which may take several forms depending on the genre. In the case of a player-versus-environment game, a scenario may be too difficult to tackle alone, requiring the aid and coordination of additional group members to fulfill specific roles. From the perspective of a competitive team-versus-team setting, the goal itself is to coordinate with your teammates to claim victory over the opposing team. In either case, by inspiring players to cooperate, game-designers open new channels of communication within the player base, facilitate different forms of game-play, and improve the game's overall appeal. Understanding the nature and impact of player behaviors in multiplayer game contexts is thus important to game growth as well as how game environments may be structured or tailored to encourage community.

The most popular and long-lasting multiplayer games have indeed cultivated their respective communities by continuously ensuring their mechanics and strategies are balanced with newly introduced content. If a game is too easy or overwhelmingly difficult, players may quickly lose interest through boredom or frustration. Dominant strategies additionally trivialize the game experience by reducing the number of interesting decisions the player can make. An appropriately balanced game therefore benefits the development of its community by fostering strategic discussion, coordination and by providing an engaging gameplay experience. Motivated by the importance of community in games (and in general), this thesis aims to show how the quantitative assessment of cooperative behavior in games can be used to isolate problem areas which may require additional balancing.

We begin by investigating techniques for measuring the level of difficulty within a cooperative player-versus-environment multiplayer game. We measure a game's "pace" as the temporal progression of its difficulty, which directly reflects the required level of cohesion and coordination among the players in a group. We focus on WORLD OF WARCRAFT (WoW) as a contemporary and popular game where player enjoyment is directly affected by the degree of difficulty experienced. As a multiplayer Role-Playing Game (RPG) with a relatively large variety of player statistics, WoW provides a rich context for measuring cooperative player experience. Player (team) health, level of cooperation and the importance of battle strategy are all potential measures of difficulty.

Using the WoW plug-in interface, we develop a basic tool for gathering real-time player data during gameplay. We explore two fundamental metrics, one inspired by LEFT 4 DEAD using team health [9], and the other a novel measure that focuses directly on the team experience, measuring cohesiveness of the group during adventuring. The latter uses a distance-based metric intended to encode the "amount" of strategy employed by the team, and thus the degree to which cooperation is important to successful gameplay.

In the second major part of this work, we examine the impact of cooperation among players in a team-based player-versus-player setting and attempt to model individual player behavior in the context of a contemporary multiplayer first person shooter (FPS), HALO: REACH. We focus here on measuring the effect of cohesive teamplay in contrast to playing as a "lone wolf". Our approach makes use of the HALO: REACH Stats API, which allows us to crawl a representative subset of the HALO: REACH player base, as well as collect a wide set of per-player gameplay metrics such as achievements, rank, total kills, total deaths, wins and score to name but a few.

We construct a weighted edge list of all the players who have played in the same game together. The edge weights of the resulting graph provide insight into the level of coordination among players. We discover that in the average case, the win-to-loss ratio of players who enter as a cohesive group into the multiplayer matchmaking system scales in proportion to the size of the group, thus alluding to the efficacy of coordinated team play in the HALO: REACH matchmaking system.

We furthermore apply an exploratory principal component analysis (PCA) to the aggregate set of player gameplay performances in an attempt to isolate specific sets of player behaviors. This step reveals a set of five stable components which can be used to describe a player's behavior and skill in relation to various game mechanics present in HALO: REACH.

1.1 Contributions

The specific contributions of this work first include the description and development of two non-invasive data collection systems which sample real-time and post-game multiplayer data in WORLD OF WARCRAFT and HALO: REACH respectively. Secondly, we analyze real-time gameplay data using two interesting metrics for understanding the extent of difficulty and cooperation in a cooperative player-versus-environment (PvE) setting; the first related to player health, and the second related to inter-player distance. We also infer coordination among player groups by isolating tight-knit components in a sampled player graph. This data shows the size and extent of player "friendships," as well as the impact of cohesive groups on game balance. Lastly, we make use of principal component analysis (PCA) on FPS multiplayer data in an attempt to categorize individual player behavior in this genre. Our results reveal interesting and novel player types strongly related to the game's mechanics.

1.2 Thesis Outline

This thesis is divided into five chapters, including this introductory chapter. Chapter 2 describes the collection and analysis of cooperative gameplay metrics in the player-versus-environment setting of WORLD OF WARCRAFT. Chapter 3 discusses the data collection and analysis of aggregate post-game data stemming from HALO: REACH's team-based player-versus-player gameplay. Chapter 4 presents relevant related work in the field of games research, with a particular focus on game metrics. Chapter 5 concludes with a summary discussion of our results, and avenues for future work.

CHAPTER 2 Measuring Cooperative Behavior in World of Warcraft

In all games, the quality of player experience is directly influenced by the level of challenge provided by the gameplay. A low level of challenge results in a trivial and uninteresting game, while a continuous and excessively high level of difficulty will result in many players abandoning the game as being too difficult. Appropriate game *pace* has thus been recognized as an important factor in ensuring players are continuously engaged without being overwhelmed [30]. Games such as LEFT 4 DEAD even include an active monitoring component that uses a combination of player measures to determine difficulty, and thus controls pace in order to match player ability [9].

In this chapter, we apply a similar pacing metric to WORLD OF WARCRAFT, specifically to multiple instances of a 5-person cooperative "dungeon". We show that a cooperative team that is well-matched to the dungeon level will exhibit clear differences in comparison to teams whose characters are over-levelled, or too powerful. In the former case, our metrics show that the difficulty level is conducive to tight teamwork. However, in the latter, difficulty is notably lower, with player characters requiring a reduced level of cohesion to complete the same content. Our results demonstrate that difficulty and pacing are clearly exposed by these basic measures.

2.1 Background

This background section presents an overview of WORLD OF WARCRAFT and the relevant game mechanics on which we base the analysis of our data. We also describe the WORLD OF WARCRAFT plugin framework, commonly referred to as client *AddOns*, which make use of the Lua-based WoW API.

2.1.1 World of Warcraft

WORLD OF WARCRAFT (WoW) is often described as a modern successor of the multi-user dungeon genre. It is currently one of the most popular Massively Multiplayer Online Role Playing Games (MMORPG), having exceeded a user base of 12 million subscribers in October 2010 [6]. WoW has undergone a drastic number of changes since its inception in 2004 including three expansions, the latest of which is WORLD OF WARCRAFT: CATACLYSM.

Leveling. The character leveling system in WORLD OF WARCRAFT closely follows the leveling systems of most RPGs. Characters start at level 1 and increase in level by gaining experience from completing quests and defeating non-player characters such as monsters, dragons, wizards, etc. The latest WoW expansion raised the level cap from 80 to 85. Note that the experiments described in this chapter were performed in the previous expansion, WRATH OF THE LICH KING, during which the level cap was set to 80. When a character gains a level, stats are incremented. Character stats include stamina, dexterity, strength, intellect, etc. The growth of these stats is prioritized according to the character's class. During the leveling process, the character is also attributed talent points, which can be spent to obtain new spells and abilities. Classes. During the character creation process, the player must select one of ten classes: Warrior, Druid, Shaman, Hunter, Paladin, Death Knight, Rogue, Mage, Warlock, and Priest. These classes follow the archetypal descriptions found in most RPGs - Warriors are melee-centric and heavily armored; Rogues rely on stealth; Mages cast destructive spells; Priests heal their teammates, etc. The majority of these classes have a unique mechanic or resource system which differentiates their gameplay. For example, Warriors use a *rage* system which accumulates over the course of a battle, allowing them to perform specific abilities. By contrast, most spell-casters use *mana* to cast spells from a distance.

Roles. A character's talent point distribution dictates his or her optimal role in a group setting. Each class has three unique *talent trees* in which talent points can be spent. Deep talents within these trees are more powerful, but also have a higher number of prerequisites. Every class has at least one talent tree which focuses on increasing combat efficiency. Warlocks, Rogues, Mages, and Hunters are pure *damage-dealers*. Warriors, Paladins, Death Knights and Druids can invest points in one of their respective trees to optimize their *tanking* ability, allowing them to mitigate incoming damage. Paladins, Druids, Shamans, and Priests each have at least one *healing* tree, which allows them to heal group members more efficiently. As such, a character can fulfill one of three roles in a group setting: tank, healer, or damage-dealer.

Cooperative gameplay. An in-game matchmaking system facilitates group formation by combining characters according to their role. Character grouping is first prioritized according to level, then according to equipment quality. One tank,

one healer and three damage-dealers are bound to an instanced dungeon, in which the group members must cooperate to defeat non-player controlled characters (NPCs). These player-versus-environment (PvE) scenarios usually take 25 minutes to complete, after which the group members are rewarded with experience, items and *dungeon points* which can be traded with vendors for more valuable equipment. Dungeons have a minimum level requirement, and can be repeated an arbitrary number of times. The matchmaking system will ensure that characters are placed in a suitable dungeon, in which the NPCs are roughly the same level as the group members. The environmental layout and events do not change between instances of the same dungeon. In most cases, enemy NPC placement and attributes also remain the same, although a small handful of dungeons spawn enemies pseudo-randomly.

When a character reaches the level cap, his or her stats can only increase by obtaining better equipment. As a level-capped character's stats increase, so does its ability to fulfill roles in more challenging dungeons, such as the "heroic" versions of previously visited dungeons. Equipment gathered from these heroic-difficulty dungeons are then used to participate in longer, more intricate raid instances. These instances can only be completed by groups of 10 to 25 players. As such, the 5-player dungeons act as stepping stones into more demanding raid environments.

One notable issue which affects the replayability of these "stepping stone" dungeons is that their content does not scale dynamically with the stats of the group members. Regardless of the players' equipment quality and stats, the strength, health, attack types, layout and number of enemy NPCs remains fixed according to the original intent of the level designers. To maximize game content replayability, the dungeon point currency system acts as an incentive for characters with comparatively higher quality equipment to revisit the 5-person dungeons, and help less advanced characters progress through the game. However, considering the static difficulty and eventual predictability of these dungeons, anecdotal player experience suggests that the gameplay becomes increasingly trivial as the character's stats grow: player health pools increase, tanks mitigate more damage, healers heal for higher amounts, and damage-dealers increase their damage throughput. Eventually, the trivial nature of higher-level player interaction with the game is perceived as mundane, or a "grind": a task whose time investment and stimulation level is unfitting with respect to the reward's perceived value.

2.1.2 WoW AddOns

WORLD OF WARCRAFT is based on a client-server architecture, wherein a game client communicates with a server to update its view of the game world. The WORLD OF WARCRAFT game client supports the use of *AddOns* (plug-ins) to customize the layout and functionality of its graphical user interface (GUI). WoW AddOns are typically written in the Lua programming language [18, 34].

The WORLD OF WARCRAFT API offers an extensive array of functions ranging from character inventory management to Mac-exclusive iTunes playback control [8]. Note that there are no functions available to automate character interactions with the game world. As such, spell-casting and character movement can only be performed as a result of the player's direct input via the mouse and keyboard. The game client contains a Lua interpreter which supports a subset of the language's libraries. Within this Lua sandbox, there is no way to directly access the underlying file system, nor is there a way to directly invoke any of the operating system's functions. Data can nonetheless be saved and loaded by means of a designated Lua save file, the name of which can be specified in the AddOn's meta-data. AddOn data is saved as Lua code, which can be run in a standalone Lua interpreter to recreate the serialized data structures outside of the WORLD OF WARCRAFT client [18].

2.2 Method

In this section, we outline the framework used to perform our experiments, as well as the metrics applied to our data. Figure 2–1 depicts the architecture of our framework. The Orunj AddOn, presented in the figure as the "Lua Script", runs in the Lua sandbox of the WORLD OF WARCRAFT game client. The AddOn records gameplay data by polling the game client for information about its view of the game state. Recorded sessions are automatically serialized by the game client into a Lua file. Offline scripts are used to analyze this data, namely by computing for each timestep in a session: (1) the total amount of health lost in the group, and (2) the minimum enclosing circle around the group. The analyzed data is exported into flat text files to facilitate plotting. It is also exported into the *.love* file format so it can be retroactively visualized through the LÖVE framework [22], a Lua-based 2D graphics library, as illustrated in Figure 2–2.



Figure 2–1: Orunj system architecture. The system we developed is identified by the blue highlight. Our data collection begins by polling the game client periodically via an AddOn written in Lua. When a session terminates, the character logs out to automatically save the data into a Lua file. In typical AddOn usage scenarios, this automatic saving procedure allows AddOns to save and reload user preferences. By contrast, we use this automated mechanism to make our data available outside the game client. We re-format and analyze the raw gameplay data by computing the health progression and Minimum Enclosing Circle around the group of players at each time interval. We also export self-contained gnuplot scripts and LOVE-compatible files to facilitate visualization and validation.



Figure 2–2: WoW positional data visualization using the Lua-based LOVE framework. The white circle (MEC) illustrates the group's position in the environment. The colored trails identify each player's trajectory and class - Pink: paladin, tank; Orange: druid, healer; Dark blue: shaman, damage-dealer; Light blue: mage, damagedealer; Red: death knight, damage-dealer.

2.2.1 Orunj AddOn

The Orunj AddOn begins its data recording procedure by instantiating required data-structures for the new recording session. An initial snapshot of the current group members is taken, including their names, their maximum health, and their maximum power (power here depends on the class' resource system such as rage, mana, runic power, energy, or focus). The talent point distribution of each character is also recorded to infer its role in the group as either a tank, healer, or a damage-dealer.

The default sampling rate is set to one sample per second to minimize memory consumption. A sample point contains each group members' health, power, and map position as an (x, y) coordinate. The map position is normalized to a unit square and represents the position of the character with respect to the 2D overhead map of the environment. Note that full 3D coordinates of the players are not accessible through the WoW API. In most cases, however, the z-coordinate of players is not significant to gameplay.

The majority of the WoW API function calls return a small subset of the local client's data, as opposed to querying the server directly. An exception to this flow is the initial character inspection process, which requires that the server be notified of which character needs inspection in order to push the appropriate data to the game client. The act of recording character health, power, and map position is thus non-invasive with respect to the server and other players, and does not perturb player actions or experience. When a session has been recorded, the player must log out of the game world to ensure that the recorded data is written to the Orunj AddOn's designated save file.

The recorded session's datastructures are loaded outside the WoW game client by means of an external Lua interpreter. Lua scripts are used to compute metrics on the collected gameplay data such as overall health loss per unit time, and the progression of the minimum enclosing circle around the group members based on their 2D map position.

2.2.2 Measuring Player Intensity

The first metric applied to the recorded data is *player intensity*, which is measured in proportion to the amount of health lost per unit time. To succeed in a dungeon, group members must cooperate to defeat enemy NPCs and avoid being killed. Similar to the *emotional intensity* metric used in LEFT 4 DEAD [9], when a group member is damaged, the *player intensity* of the group is incremented proportionally. For each sample in a recorded session, player intensity is computed by summing the total amount of health lost by each group member, divided by the total health of the group. This yields a normalized value between 0 and 1 for each sample, which allows for comparisons of player intensity metrics between groups.

Periods of high intensity correspond to moments in which group members sustain a large amount of damage (i.e. health loss) per unit time. For a group to succeed, its players must meet some baseline level of cooperation to survive moments of high intensity. As intensity increases, so should the level of involvement required by each player to stay alive. This level of involvement contributes to the gameplay experience by promoting the player's focus and motivation. If the amount of intensity is insufficient, the scenario becomes trivial and the players may lose interest. Of course, continuous high-intensity scenarios may also prove to be too stressful and too challenging, hence the importance of appropriate pacing. Such proper pacing is generally marked by periods of high intensity separated by lulls in the action to ensure the players are not overwhelmed.

It is worth noting, however, that other game genres such as puzzle games or social games may benefit from a more relaxed pacing. In such cases, player intensity may not be an appropriate metric to gauge player involvement.

2.2.3 Minimum Enclosing Circle Progression

The second metric applied to the collected data is the progression of the minimum enclosing circle (MEC) around the group. The MEC algorithm [28] takes as input the 2D map position of each character in the group, and returns the (x,y) coordinates of the MEC's center, as well as its radius.

To justify the relevance of the MEC metric, we first present a set of key observations in relation to WoW's gameplay mechanics:

- Spell-casting relies on the character's distance and line of sight from its target. Ranged damage-dealers must generally stay within 35 yards of their target, while healers can heal from a distance of up to 40 yards. Melee classes must stay within melee range of their target.
- Most spell-casting requires that the character be stationary. Melee classes maximize their damage throughput when they are standing behind their

target, to reduce the chance of parries or blocks. As such, the tank should avoid moving the enemy NPC's unnecessarily.

• Healers and damage-dealers tend to stay behind the tank as it leads the group through the dungeon; the tank is generally the first group member to engage in battle.

As team cohesion increases, a smaller MEC radius is maintained with most players positioned within 35 to 40 yards of one another. This way, the healer can continuously heal the group members while the damage dealers focus on defeating the enemies. A much larger MEC radius is expected in cases of low cohesion. It is often the case that one or more freeloading group members trail behind when the scenario's difficulty is made trivial by overpowered players.

To complete a dungeon, the players must defeat groups of enemy NPCs along a generally linear path towards the dungeon's final boss. In appropriately challenging scenarios, each NPC encounter will last long enough to have each group member participate in the fight, for example by casting spells. As such, a heatmap of the MECs is expected to display tight overlapping circles for prolonged NPC battles. The speed and position of the MEC's center should only change as the group moves to the next battle. It is therefore expected that an easy scenario would be characterized by a highly mobile MEC center; the group would spend less time *engaged* in battle, and more time *moving* towards the final boss.

To summarize our metrics, the pacing of a scenario is expected to match moments of high and low player intensity, as well as the temporal progression of the MEC's overlap and radius.

2.3 Experimental Results

In this section, we describe the experiments performed in WORLD OF WARCRAFT's dungeon matchmaking system. We also analyze our results and discuss their relevance towards measuring perceived intensity and cooperative pacing in WoW.

2.3.1 Experimental Context

Three characters were used to perform the experiments reported in this chapter, namely: (1) "Appuls", a level 80 Druid with high-quality equipment obtained from 10 and-25 person raids, (2) "Orunjs", a level 80 Death Knight (not to be confused with the name of our *Orunj* WoW AddOn), with comparatively lower quality equipment gathered from quests and 5 person heroic dungeons, and (3) "Aid", a level 70 Priest equipped with questing items. Appuls and Aid had their talent point allocation optimized for healing, whereas Orunjs was optimized for tanking.

Constraints in the data gathering phase required author participation, which introduces significant potential for bias. This was done to ensure that the character collecting the data was within range of each other character to consistently record their health and positional data. In a healing role (playing "Aid" and "Appuls"), the author had a low influence on player intensity, as there was no way to control how much damage the other group members would take from enemy NPCs. In a tanking role (playing "Orunjs"), the player intensity metric is subject to a higher potential for bias; the tank is partly responsible for ensuring that enemy NPCs do not hit other group members. Regardless of the character role, the author's influence on the MEC path is quite low, given the linear environmental layout of most dungeons. To reduce this potential for bias in future work, a larger number of data gathering participants can be used to collect and upload their data to a central repository for later analysis.

We examined the results from the regular and heroic versions of the "Utgarde Keep" dungeon (see Figure 2–3). Aid was used to collect the results of the regular version of Utgarde Keep - the regular version of Utgarde Keep is the first WRATH OF THE LICH KING dungeon available at level 70. Appuls and Orunjs were used to collect data from the heroic version of Utgarde Keep, accessible at level 80.

In terms of character progression in WRATH OF THE LICH KING, a level 70 character, such as Aid, is normally equipped with items whose "item level" (ilvl) is valued at 138. The majority of these items are collected from quests in the level 70 questing area. A freshly-attained level 80 character, such as Orunjs, is generally equipped with items ranging in value from ilvl 160 to ilvl 200. By contrast, a level 80 character who regularly participates in 25-person raids, such as Appuls, is usually equipped with items ranging in value from ilvl 226 to 264. Item level is proportional to the amount of allocated stats on an item, thus increasing the efficiency and power of the character as his/her items increase in ilvl. A mapping of item level to character progression can be found at [35].

Recall that the dungeon matchmaking system prioritizes group formation according to level, then equipment quality. Aid's groups consisted of characters ranging from level 69 to 72. Appuls' and Orunjs' groups consisted of level 80



Figure 2–3: *Playable area of the first floor of Utgarde Keep.* Players enter the dungeon on the bottom right, and proceed to the second floor of the dungeon, following a horseshoe-shaped path.

characters, though Appuls' group members usually had higher stats, imparted by higher-quality items.

A number of runs were recorded throughout the development process and testing of our framework. The majority of these runs served to ensure that the data collection logic could robustly detect and handle map changes, in-instance floor increments (eg: going up stairs occasionally changes the map), and player replacements in the group (eg: a player leaving and being replaced by an entirely different character). We also performed diagnostic runs of several dungeons to examine the readability of the plotted data on certain maps. Indeed, maps such as Violet Hold which forced players to move back and forth between NPC spawn locations were not conducive to producing readable and easily interpretable positional data (see Figure 2–4).

Obtaining usable data from dungeon runs also proved to be challenging. In normal play conditions, it was not unusual to have one or several party members leave the group, either due to a disconnection, frustration, or real-life obligations. This imposed significant down-time as the remaining party members attempted to salvage the situation and find replacements. This down-time would later translate into a disproportionately large overlap in the MEC heatmap, as the players maintained their position due to their lack of ability to progress in the dungeon without a full group.

The readability of the MEC plots is furthermore affected when a character dies during the run. The party must typically wait for him or her to revive him/herself at the dungeon entrance, and run back (or be summoned) to the group. When a



Figure 2–4: Player positional data from the Violet Hold dungeon. This dungeon requires that the players move back and forth between pseudo-random NPC spawn locations. The resulting plot of this dungeon is not easily interpreted, and was therefore not selected for our experiment.

character is distant in the environment, the local game client does not consistently receive position updates since both characters are outside each others' spheres of influence. The plot therefore reports the revived character's movement as a series of "jumps". The resulting MEC heatmap moreover suffers from extremely large overlapping circles which reduce in size as the revived player(s) approach the group (see Figure 2–5).

To mitigate this, we introduce an arbitrary threshold value in which players are ignored if they are too distant from the previous timestep's MEC center. As their position crosses the threshold value, it is included in the next MEC calculation, usually resulting in a sudden discontinuous jump in the MEC's radius. To remove this artifact in future work, a post-processing data filter may be implemented to ensure a continuous MEC progression.

2.3.2 Intensity Results

Aid. The regular version of Utgarde Keep was designed and balanced for characters between levels 69 and 72. The data collected with Aid therefore serves as a basis of comparison for the intended pace and intensity of this dungeon. Figure 2–6 (a) illustrates the player intensity for a playthrough of the regular version of Utgarde Keep. Observe that the player intensity ramps up during each battle, and then quickly diminishes as the enemy NPCs are defeated.

There are approximately 20 NPC battles in a typical session of Utgarde Keep, including three boss fights. In Figure 2–6 (a), the first boss fight ends just before the 400 second mark. The enemy NPC encounters become increasingly challenging, causing larger spikes in the group intensity. The second boss fight occurs at around



Figure 2–5: Player position and MEC plot in a sample Utgarde Keep run. The MEC heatmap (in orange) suffers in readability from the large overlapping circles as revived player(s) make their way back to the group. The long straight lines denote the infrequent position updates of a distant player whose sphere of influence does not intersect that of the local client. Note that this is a raw representation of the positional data, which is inverted along the y-axis in comparison to the remaining Utgarde Keep MEC plots in this chapter.

800 seconds. The periods of 0 intensity correspond to group movement towards the next battle. It is worth noting that the intensity peaks at the end of the playthrough, when the group must defeat the last boss of the dungeon. This intuitively serves as a culminating point for player intensity, in which the group can lose upwards of 25% of its maximum health from a single attack.

Orunjs. Like most heroic dungeons in WRATH OF THE LICH KING, the heroic mode of Utgarde Keep was balanced to challenge characters like Orunjs, who had just attained level 80 and who were primarily equipped with items obtained by leveling. Orunjs was often paired with similarly equipped players by the dungeon matchmaking system. Thus, the player intensity in Figure 2–6 (b) shows that Orunjs' session through the heroic version of Utgarde Keep matches the level of intensity recorded by Aid's group in Figure 2–6 (a).

Appuls. Figure 2–6 (c) illustrates the intensity of a heroic Utgarde Keep session recorded by Appuls, a character equipped with raid-quality items. The player intensity in this session is distinctly lower than that of Figures 2–6 (a) and 2–6 (b), which supports the intensity metric predictions of Section 2.2.2. The duration of the session was also substantially lower than the previous sessions: 796 seconds compared to 1429 and 1295 seconds for Aid's and Orunjs' playthroughs respectively.

The higher stats of Appuls and his group members, imparted by their higher-quality raid items, contributed strongly to the reduction in player intensity in heroic Utgarde Keep. In comparison to Orunjs' and Aid's groups, more damage



Figure 2–6: *Player intensity* in regular Utgarde Keep. Each plot represents one recorded session. The sequence number along the x-axis represents the elapsed time in seconds. The player intensity is defined as the cumulative health loss of each group member divided by the total group's health at each timestep. (a) Aid, level 70, quest-reward equipment. This intensity plot serves as a point of comparison for the intended level of difficulty for this dungeon. (b) Orunjs, level 80, dungeon and quest reward equipment. The intensity pattern is similar to the one recorded by Aid. The stats and skill of the participants is fitting of the dungeon's difficulty; the heroic version of this dungeon was designed for freshly-attained level 80 characters such as Orunjs, similarly to how the normal version of this dungeon was designed for freshly-attained level 70 characters such as Aid. (c) Appuls, level 80, end-game raid equipment. The lack of intensity here suggests that the stats and skill of the group members far surpasses the dungeon's difficulty.
was mitigated, character health pools were larger, and the damage and healing throughputs of each player were greater.

The reduction in player intensity for Appuls' session thus attests to the relevance of the player intensity metric when attempting to quantify the level of difficulty to which the player is subjected. To build on this observation, a more thorough experiment can be constructed whereby the summed or averaged item level (ilvl) of the group member's equipment can be correlated with the intensity distribution throughout various dungeons. Unfortunately, the current data set collected in this study does not contain each group member's equipment list, and so the item levels are inaccessible.

2.3.3 Pacing Results

The MEC metric was used to compare gameplay pacing between the sessions recorded by Aid and Appuls. For clarity, we report the pacing results of the first floor of the Utgarde Keep dungeon. Note that the plots in Figures 2–7 (a) and 2–7 (b) correspond to the trajectory suggested by the overhead map of the dungeon in Figure 2–3.

Figures 2–7 (a) and 2–7 (b) represent the cumulative MEC data of three sessions collected by Aid and Appuls respectively. Table 2–1 illustrates an aggregate of the gameplay data used to generate Figure 2–7. These plots can be viewed as heatmaps of the average amount of time spent by each group in a specific area of the dungeon. The regions of the plots darkened by the superimposition of MECs correspond to areas in which the groups spent more time. The lighter regions of the plots denote areas of group movement towards the next battle.



Figure 2–7: (a) Cumulative MEC data of 3 sessions in regular Utgarde Keep (first floor) recorded by Aid. Darker regions indicate areas in which groups spent more time battling enemy NPCs. (b) Cumulative MEC data of 3 sessions in heroic Utgarde Keep (first floor) recorded by Appuls. The lighter regions of this plot allude to a much quicker pace.

Aggregate Metric	Aid	Appuls
MEC Speed		
Average	0.0025	0.0042
Standard Deviation	0.0037	0.0062
MEC Radius		
Average	0.0210	0.0258
Standard Deviation	0.0090	0.0213
Intensity		
Average	0.0074	0.0056
Standard Deviation	0.0181	0.0144
MEC Speed vs Intensity		
Pearson Correlation	-0.1308	-0.0413
p-value	2.4116e-13	0.0953

Table 2–1: *MEC Metrics and Intensity*. Quantitative metrics presenting the average MEC speed and radius, and the intensity results for Figure 2–7.

The MEC heatmaps recorded by Aid in Figure 2–7 (a) for regular Utgarde Keep are noticeably denser than the ones recorded by Appuls in Figure 2–7 (b) for heroic Utgarde Keep. The "MEC Speed" and "MEC Radius" rows of Table 2–1 attest to this difference in density; the average speed of Aid's groups is 40% slower than that of Appuls' groups, causing more MEC overlaps in the heatmap. The standard deviation of the MEC radius is also greater in Appuls' groups, suggesting there is more variance and therefore less consistent group movement in Appuls' groups, compared to Aid's groups.

The intensity metric in Table 2–1 is also worth comparing. The average intensity in Appuls' groups is approximately 25% lower than that of Aid's groups, while maintaining a similar standard deviation. This alludes to the higher damage

mitigation and better items of Appuls' group members versus Aid's group members, relative to the dungeon NPCs' attack strength.

The correlation between the MEC speed and the group intensity is also revealing. In the case of Aid's groups, the Pearson correlation value of -0.1308 indicates that the MEC speed tends to be lower in moments of higher intensity. The associated p-value of 2.4116e-13 indicates a low probability of an uncorrelated system producing datasets having a Pearson correlation at least as extreme as the one computed. By contrast, the Pearson correlation of -0.0413 for Appuls is closer to 0, and its p-value is much larger, suggesting that in Appuls' groups, the MEC speed apparently varies independently from the group intensity. This can be justified by the group's high mobility engendered by the lack of a need to stay stationary against groups of enemy NPCs (the NPCs die too quickly).

The differences mentioned above support the idea that the groups in which Aid participated spent more time in each battle, and therefore followed a more progressive pace than Appuls' groups, whose individual battles were much shorter in duration. This disparity in pacing can be explained by the static difficulty of the dungeons in WoW. As characters obtain better equipment from (1) the dungeon point system, or (2) the 10 and-25 person raids, they become notably more efficient at completing 5-person dungeons such as heroic Utgarde Keep. The trivialization of this content may, in the long run, cause players to become apathetic to the gameplay which would subsequently hinder the game's replayability.

29

2.4 Discussion

MMORPGs rely on continuing and long-term player participation, typically fostered by a growth or enhancement system for player avatars. Ensuring the game remains relevant to players despite their increase in abilities and resources is thus important to player retention. Unchecked or unmatched by sufficiently abundant new and higher-level content, however, such "power-creep" has potential to trivialize game challenges, making the game experience uninteresting. Avatar level restrictions are often applied to avoid this, but especially in the context of cooperative multiplayer scenarios, difficulty can be seen as a complex function of group balance, resources, and relative player skill.

More direct measures of player difficulty can allow for a more straightforward evaluation of the extent and pacing of challenge, and thus the amount and relative level of player attention. Our results here demonstrate that easily gathered data can successfully summarize group experience, illustrating both the degree of difficulty and the how well it is distributed during actual gameplay. Additional data collected with these tools can help us further understand cooperative multiplayer behaviour and reactions to specific game content, as well as in developing adaptive systems that may dynamically adjust the difficulty to match the group ability. Such systems would notably increase the need for cohesion and communication among group members, thus fostering a stronger sense of community and inter-dependence among the player base.

CHAPTER 3 Measuring Cooperative Behavior in Halo: Reach

Our approach in this chapter makes use of the growing trend by game studios to offer web-portals for users to display and compare personalized gameplay data; examples of such portals include Battlefield 3's Battlelog [14], the WORLD OF WARCRAFT home page [7], the Bungie.net Stats page [11] and the Steam Community page [33]. Although intended to foster game community [29], these systems include a wealth of user data, and so provide interesting opportunities to analyze large volumes of real-world gameplay data [21, 16].

Along this vein, we make use of the HALO: REACH Stats API to crawl data obtainable via the Bungie.net Stats page. We focus our attention on HALO: REACH, a first-person shooter (FPS) whose relevance in the genre is reflected by its classically-rooted game mechanics, and whose popularity is stated by its 1.3 billion games played within its first four months of release [10, 1].

The methodology and analyses in this chapter differ from the previous chapter's; we focus here on the collection of of post-game statistics instead of polling real-time data. Moreover, our analysis investigates the effect of cooperative behavior in a player-versus-player environment rather than a player-versus-environment setting. Lastly, in light of the significantly larger data set described in this chapter, our results provide a much more global perspective of cooperative behavior, in lieu of a fine-grained case-study approach. This chapter begins by providing a summary background of HALO: REACH and its relevant game mechanics. We then describe our data collection process, and follow with the analysis of the player graph and an exploratory principal component analysis of the player performance statistics. We conclude this chapter with a brief discussion of our results.

3.1 Background

This sections provides an overview of HALO: REACH's gameplay and presents the basic functioning of the underlying Xbox Live matchmaking system. We also familiarize the reader with a match's post-game statistics, to help better understand our later analysis. Lastly, we describe the technical components of the HALO: REACH API and its limitations.

3.1.1 Halo: Reach

HALO: REACH is a recent installment of the HALO franchise, which has established itself as a popular series in the console FPS genre [27]. The game features a single-player campaign, as well as an array of multiplayer modes including campaign coop, free-for-all and team games. The measurements performed in this chapter only pertain to competitive team-based game types; in this context a match-making system allows players to form teams, with results from the ensuing gameplay published to the post-game lobby.

Matchmaking. The HALO: REACH matchmaking system allows players to be matched with one another according to a desired *playlist*. A playlist in this matchmaking system can be understood as a set of possible game modes. For example, the "Team Objective" playlist organizes players into two teams of size 4 and places them into an objective-based game type such as capture-the-flag or king-of-the-hill. Larger team sizes are also permitted in other playlists, such as "Big Team Battle," which pits two teams of up to 8 players against each other.

The matchmaking system is provided through the Xbox Live network service associated with the console. This service allows players to organize and maintain lists of friends, which can then be used to join the HALO: REACH matchmaking system as a cohesive group, maximizing the ability of a set of friends to play together. The service also provides persistent player identities through a unique gamertag assigned to each service account. It is possible to play with up to three guests on the same console, in which case these players are identified as guests of the host gamertag. Guests are restricted to playing in non-competitive playlists, and as such, our metrics do not encompass guest gameplay data.

Outside of group entry, the matchmaking system uses the Xbox Live "TrueSkill" ranking system to match players of similar skill together [24]. In contrast to the well-known ELO rating system used in Chess, the TrueSkill system makes use of a Bayesian update mechanism to estimate a lower bound on a player's skill, iteratively narrowing the associated uncertainty of this skill estimate from repeated gameplays [17]. The TrueSkill ranking system updates its results based on the final outcome of the game and operates under the assumption that this outcome reflects the skill value of each participant. The TrueSkill system then matches teams according to the summed skill-rating calculated for each team.

Gameplay. The most popular and competitive playlists involve two teams of four players, for a total of eight players in a game. Each player then begins by

selecting a preset *loadout*, which dictates his or her two starting weapons. Loadouts also define the player's special ability such as: invisibility and radar-jamming, sprinting, jump-jetting, temporary invincibility, and dodging. Loadout presets can also be changed while the player is respawning.

Highly competitive playlists usually provide players with a default medium-range rifle capable of precise headshots regardless of the selected loadout. Players can only carry up to two weapons at a time, which can be swapped for more powerful weapons found on the game-map, such as sniper rifles and rocket launchers. Players also have shields which limit damage taken, but must recharge a few moments after taking damage. If a player's shields are depleted, damage will be taken by enemy fire and must be regenerated by consuming medkits found on the map; shieldless, several weapons also become capable of dispatching him or her with a single headshot. A player who is killed is temporarily removed from gameplay, and must wait before respawning. The game ends if a team obtains the required number of kills, or if the objective has been captured a set amount of times, or if time runs out.

Post-game lobby. Once a game finishes, the players are placed in the post-game lobby. A summary of each player's performance is presented here, which includes his or her number of kills, deaths, assists, betrayals, headshots, as well as a detailed account of the player's earned medals, derived from achieved headshots as well as many other possible player feats, such as double-kills, assisted kills, and killing sprees. Medals do not have a direct in-game effect, and are mainly used to encourage competitive social display and provide players with more difficult and

long-term goals. As we will show later, they can also serve as a means of measuring and understanding behavior. Medals are broken down by the game system into four basic categories:

- Multi Medals Awarded for a series of kills within four seconds of each other.
- Spree Medals Awarded for a number of kills in a row without dying.
- Style Medals Awarded for feats, such as headshots, assists, and assassinations.
- Other Medals Awarded for additional feats, such as sniper-rifle headshots, melee kills, or hitting an opponent with a vehicle.

In addition to the Xbox Live friends list, players who enjoyed playing together can opt to form a party from the post-game lobby. This mechanism allows for new, online friends to be discovered, and these newly-grouped players can continue playing on the same team in subsequent matches formed by the matchmaking system.

3.1.2 Stats API

Individual HALO: REACH game statistics are hosted on the Bungie.net Stats page. For each game, users can browse the information provided in the post-game lobby and can view a temporal progression of events on the map.

The majority of this data can also be accessed via calls to the HALO: REACH Stats API, allowing development-oriented players to aggregate or explore the data. To use this API, a valid Xbox Live account is required, along with a subscription to the Bungie Pro service. The main purpose of Bungie Pro is to provide users with the functionality to record, render and download high-quality, in-engine videos. Bungie Pro also supplies developers with API keys to pipe HALO: REACH game statistics through their web servers or applications. The endpoints of the HALO: REACH Stats API are compatible with PHP and .NET. Client applications written with the .NET framework must reference a Windows Communication Foundation (WCF) service to obtain the API class definitions; in either case a valid API key must be included with each method call, and a rate limit of 300 requests per minute is enforced.



Figure 3–1: Halo: Reach multiplayer data analysis system.

3.2 Data Collection

In this section, we outline the framework we designed and used to gather and process player data. The overall design of our data gathering framework is illustrated in Figure 3–1. A program written in C# uses the .NET endpoint of the HALO: REACH Stats API to gather post-game player data, which is then cached for processing by network and statistical analysis libraries.

Discovering random player names is difficult, and so our approach is based on crawling through players connected by common gameplay. A round of data-collection begins from a chosen seed player, whose gamertag is added to a *visited players* list. This introduces a potential bias from the choice of seed, which we address below by evaluating multiple data sets.

We inspect a player's 75 most recent competitive team games, assigning unique game-IDs to avoid duplicate game inspection. The choice of 75 games is based on the number of games returned per API query. A query returns a maximum of 25 games, and so three queries per player provides a reasonably sufficient number of games for each player. Obtaining all the games ever played for each player may otherwise cause some players to have a disproportionately large weight in our analysis, specifically if they play often.

We enqueue newly-encountered gamertags found in each game and update an undirected, weighted edge list to identify the players who have played together in the same game. We also inspect each player's performance per game, defined by a row vector containing his or her score, team standing, individual standing, kills, deaths, assists, suicides, betrayals, number of medals earned per category, unique medals earned per category, average death distance, average kill distance, and number of headshots.

The next iteration pops a gamertag from the queue and adds it to the list of *visited players*. We repeat the process of game inspection, edge-list maintenance

and game performance caching for each player added to the *visited players* list. HALO: REACH is a popular game, and so continuing this process until a closed set is found is not feasible, at least not given the rate-limits imposed by the API.

The data-collection program is halted manually after approximately four days of execution. We do this to ensure that the constructed network is large enough to be statistically significant, but small enough to have Python's NetworkX functions terminate within a reasonable timeframe. For reference, a sample set constructed from 2 443 visited players and containing 3 812 330 edges (as illustrated in Round 3 of Table 3–1) takes approximately two days to analyze with our various Python scripts. As we describe later, despite the differences in size between our four rounds of data collection, the network metrics and principal component analysis reveal consistent trends throughout these datasets.

The Python NetworkX module is used to construct a player graph from the weighted edge list [26]. Network metrics are calculated on this graph to plot node-degree distribution, edge-weight distribution and average component size. We also aggregate the cached gameplay performances of each player to compute his or her average performance per game.

We make use of the R programming language, specialized to facilitate statistical computations, to run a principal component analysis (PCA) on this data. We ensure the algorithm follows a singular value decomposition on the centered and uniformly scaled column values [19]. The resulting output is a list of orthogonal vectors (principal components) whose linear combination can be used to describe the individual performance of each player according to a new basis. Each principal component accounts for a specific proportion of the variance in the input data. The use of PCA in this experiment aims at reducing the number of initial variables used to qualify a player's performance (score, team standing, individual standing, kills, etc) into a smaller set of correlated variables.

3.3 Data Analysis

The results analyzed in this section are based on four rounds of data collection, yielding a total of 6700 unique visited players and 384124 uniquely inspected games. The first three rounds of data collection follow a breadth-first search from three different seed gamertags. The fourth round of data collection uses the same seed as the first round of data collection, but a random walk is used instead of a breadth-first sampling approach. We present the volume of data collected for each round of data collection in Table 3–1. Our concern is that a breadth-first crawl would yield a biased set of visited players alike in skill and behavior. However, the similarities between network topologies and PCA results support the equivalence of both sampling methods, which is illustrated by the similarities in Figures 3–3, 3–4 through 3–7, and 3–8.

3.3.1 Player Graph

We construct an undirected graph from the player edge list. The set of nodes in this graph corresponds to the set of all players encountered in the inspected games. The edges in this graph represent the fact that two players have played in the same game, with the weight of the edge denoting the number of times a pair of connected players have played games together.

Round	Crawl Technique	Visited Players	Number of Edges	Visited Games
1	Breadth First	787	1 596 832	47 687
2	Breadth First	2 156	$3\ 623\ 027$	119 846
3	Breadth First	2 443	$3\ 812\ 330$	140 048
4	Random Walk	1 352	2 750 236	79 069
merged	-	6 700	17 457	384 124

Table 3–1: Rounds of data collection. The fourth round of data collection uses a random walk to crawl the player base using the same seed name as round 1. The merged data set contains a notably reduced number of edges; we only conserve edges of weight 8 and over to maximize the speed of the merging process, and to ensure that the Python NetworkX algorithm used to count components in a graph can terminate in a timely manner. Observe that the number of Visited Players in the merged data set (6 700) contains slightly less players than the sum of all Visited Players (6 738), indicating a small overlap. This overlap originates from the use of the same seed name for rounds 1 and 4.

Our data crawl is artificially truncated to maintain feasibility of the crawling process, and so not all players we encounter are fully explored. As illustrated in Figure 3–2, a node in the graph can thus belong to one of two sets:

- 1. Visited Players: the set of players whose recent games have been inspected and used to further crawl the player base.
- 2. Satellite Players: the set of players who have appeared in at least one inspected game, but who have not been included in the set of visited players.

Figure 3–3 presents the node degree distribution of the player graphs in our four rounds of data collection. Table 3–2 provides a quantitative comparison of this node degree distribution and reveals a similar fit across all four rounds of data collection. The number of nodes in these graphs far exceeds the number of visited players. For example, our largest data set, Round 3, encompasses 2 443 visited players, 411 877 nodes, and 3 812 330 edges. The sharp peak on the left of these



Figure 3–2: A *conceptual simplification* of the sampled player graph; the green nodes represent the visited players, while the blue nodes represent the satellite players.



Figure 3–3: Node degree distribution of Rounds 1 through 4 of data collection. The sharp peak on the left corresponds to the numerous satellite players who are connected to at least one visited player. The green and blue histograms have been individually normalized, such that the integral over their respective ranges is 1. The value of each bin is the result of the probability density function at that bin. Note that the y-axis is presented on a logarithmic scale in these plots.

Round	Avg. Node Degree	Std. Deviation.	Avg. Node Degree	Std. Deviation.
	(visited)	(visited)	(satellite)	(satellite)
1	439.85	192.72	12.27	13.20
2	396.30	157.32	14.33	15.01
3	399.52	147.36	16.24	18.03
4	405.81	168.31	14.72	17.65

Table 3–2: *Node Degree Distributions*. In the table above, we can observe similar parameters for the fitted gaussian distributions in both the visited player and satellite player node degree distributions.

plots corresponds to the node degree of the satellite players, while the hump near the middle corresponds to the set of visited players. Each visited player is necessarily connected to each player appearing in his or her inspected games, while each satellite player is connected to at least one visited player in addition to the other players appearing in the same inspected game(s). A satellite player can be connected to more than one visited player, in particular if he or she is friends with several visited players.

Figures 3–4 through 3–7 present the edge weight distribution of the player graphs in our four rounds of data collection. As expected throughout these figures, the mean edge weight for pairs of satellite players is approximately 1, with a standard deviation between 0.45 and 0.51. By comparison, the mean edge weight between pairs of satellite players and visited players increases slightly between 1.18 and 1.21, as does its standard deviation bound between 1.24 and 1.61. This larger standard deviation is caused by a higher number of occurences of larger edge weights. This fact is important to keep in mind, as later in our analysis we will denote two pairs as "friends" if their edgeweight surpasses a given threshold. Such friendships can exist between satellite players and visited players, hence why we must keep the satellite players within our dataset.

The mean edge weight between pairs of visited players is larger, varying between 2.49 and 3.00, and with a wide-spread standard deviation between and 7.97 and 9.81. This observation is accounted for by how we crawl through the dataset; the set of one visited player's games must overlap at least once with another visited player's set of games. If these two happen to be friends, the size of the intersecting sets will naturally increase.

By construction, the player graph is initially defined as one large component. That is to say, there exists at least one path from one node to every other node in the graph. To evaluate the impact of cooperation among the players in this data set, we begin by identifying the groups of players who play together often. We simplify the terminology by labeling a pair of players as "friends" if the two have played in at least eight games together. This threshold is justified by an analysis of component size built by progressively pruning edges of increasing weight. By pruning edges of increasing weight, we fragment the network into smaller components composed of what we may assume as individuals who have chosen to play together; given the size of the HALO: REACH player-base, and the number of games sampled per player, it is unlikely that two strangers will be placed by the matchmaking system into the same eight games regardless of their similarity in skill rating. The results are shown in Figure 3–8, and strongly indicate a component size stabilizing point of around eight. Since players can opt to continue playing together





Figure 3–4: *Edge weight distributions in Round 1 of the data collection.* We illustrate the various edge weight distributions between: (green) pairs of satellite players; (red) pairs of satellite and visited players; (blue) pairs of visited players.



Figure 3–5: *Edge weight distributions in Round 2 of the data collection.* We illustrate the various edge weight distributions between: (green) pairs of satellite players; (red) pairs of satellite and visited players; (blue) pairs of visited players.





Figure 3–6: *Edge weight distributions in Round 3 of the data collection*. We illustrate the various edge weight distributions between: (green) pairs of satellite players; (red) pairs of satellite and visited players; (blue) pairs of visited players.



Figure 3–7: *Edge weight distributions in Round 4 of the data collection*. We illustrate the various edge weight distributions between: (green) pairs of satellite players; (red) pairs of satellite and visited players; (blue) pairs of visited players.

by forming a party in the post-game lobby, our use of the word "friend" does not necessarily denote a friendship in the Xbox Live service. As evidenced by the capacity to reliably enter the same game, friendship implies a level of coordination between two players. We assume that such a friendship also implies a willingness to cooperate.

Our interest in identifying friendship groups is based on a hypothesis that friendship has an impact on team success in competitive situations. Since friendships can be formed based on successful, randomly-formed competitions, the causal direction of such a relation is not trivial to disentangle, but is interesting from either perspective as a potential factor in game design and balancing. Friends can enter the matchmaking system as a party of up to eight players, although the majority of competitive playlists only allow a maximum party-size of four. For each visited player in our merged data set, we compare his or her wins ratio with respect to the number of his or her friends in the game. Figure 3–9 shows the increasing progression of the average win ratio in relation to the number of friends in the game. In the median case, the win ratio is proportional to the number of friends in the game.

We posit that the median and lower quartile dip at x = 4 and x = 5 is explained by the jump from playlists only allowing a maximum of four players per party, to those accepting a maximum of eight players per party. The hypothesis would follow that with four or five friends involved, a team must be composed of eight players, but is not yet fully dominated by friendships, and so still requires significant and necessarily suboptimal coordination with strangers. Showing this



Figure 3–8: Average component size versus pruned edge weight, Rounds 1 through 4. The graph begins as one large component. As edges are pruned, the average component size quickly stabilizes to a value less than 4. Based on these results, we choose a weight threshold of 8 to qualify player pairs as friends.



Figure 3–9: Average Win Ratio Versus Number of Friends In Game. In this box plot, the x axis identifies the number of friends in the game. The red line represents the median win ratio associated with the number of friends in the game. The blue box surrounding the median extends from the lower to the upper quartile of the win ratio values. The whiskers show the range of the data. The increasing trend of the upper quartiles as well as the increasing median illustrate the strong benefit of repeated coordination with friends.

would require more data about the maximum party size for each game visited, which we leave for future work.

3.3.2 PCA

The second part of our experiment pertaining to this chapter aims to isolate and categorize individual player behaviors found in our sample. Competitive, team-based play encourages aspects of cooperation, but also provides the potential for different play styles through the variety of available loadouts and weapons, such as stealthy guerilla tactics, long-range sniping or close-range assault. It is also worth noting that the real-time and dynamic nature of multiplayer gameplay does not necessarily allow players to distinctively exhibit their preferential play styles—a failed assassination cannot be retried by reverting the game to a previous state, as is the case with most single player games. Thus, a strict adherence to a specific play style may be discouraged in favor of coordinated team play.

Our analysis is based on the average performance per game of each visited player. We considered as much of the individual data available per player as possible, applying a principal component analysis as detailed in Section 3.2. We present the simplified PCA results of each round of data collection in Tables 3–3, 3–4, 3–5, and 3–6, as well as the PCA results of the merged data in Table 3–7. In these tables, we conserve the sign of the coefficients whose absolute value is greater than half of the absolute value of the largest coefficient. Opposite signs indicate opposite variable correlation. The percentage beneath each principal component (PC) represents its proportion of variance. Bracketted signs indicate an exceptionally large absolute value for a coefficient and thus a dominant behavior. Each round of data collection yielded a set of five or six principal components whose standard deviation was above 1, the conventional cutoff value [19]. For the sake of comparison, we only consider the first five principal components across all the rounds of data collection. Observe that the distribution of the proportions of variance are similar among these principal components, and that all the data sets are dominated by a component which describes a player's aptitude at obtaining kills and medals. The similarity in the PCA results attests to the equivalence of the breadth-first versus random-walk sampling methods used in our data collection phase. A more fine-grained analysis of these PCA results is provided in their respective captions.

The analysis of the merged data yields a set of five major principal components whose cumulative proportion of variance is valued at 79%. In other words, the weighted linear combination of these five vectors accounts for 79% of the variance among the 6 700 player performances.

Players who have a large **PC1** weight show an overall aptitude for obtaining kills and medals, attesting to their comfort and versatility in varied gameplay scenarios.

A large negative weight along **PC2** corresponds to a player's ability to perform headshots. It is worth noting that this variable negatively correlates with assists. Intuitively, players who achieve many headshots are awarded with the killing blow and not the assist. The same-sign correlation between headshots and kill/death distance is explained by the longer effective range of weapons capable of headshots. Players with a positive weight for **PC3** are better suited to obtain assists, while a negative weight along **PC4** indicates a disposition towards long-range engagements. In this later component, the average kill/death distance appears to be correlated with the spree medals, but oppositely correlated to score. Note that objective-based games increase a player's score according to each objective captured and not according to his or her number of kills. As such, players who are far removed from the action are also less likely to score objective points.

The Standing metric is a binary ranking (lower values meaning higher ranks), and indicates whether the player's team has won (Standing = 0) or lost (Standing = 1). **PC5** is focused on this metric. Thus, a player with a large negative value along **PC5** indicates that the player's team loses more often (Standing closer to 1). By contrast, a player with a more positive weight for **PC5** indicates that the player's Standing is closer to 0, and so implying that his or her team wins more often. In **PC5**, Standing is also correlated with Individual Standing, a more fine-grain ranking value that measures the player's overall performance in a game compared to the other player scores, regardless of team. An Individual Standing of 0 indicates the first-place position, while a value of 7 denotes the last place position in a game of 8 players. Observe that in **PC5** these two measures are both correlated, and so players whose teams lose more often (Standing closer to 1) would tend to have larger Individual Standings (ie worse, or larger individual rankings).

The correlation of the variables within the principal components reveals that player types roughly correspond to the use of the available game mechanics of HALO: REACH. This is in contrast to broader player types found in other game contexts [5, 2, 36], and in partial contradiction to our initial expectations that both assistive "heart" types and traditional "griefing" or "club" behaviours [5] would be the most strongly represented. Game types are instead dominated by skill in general (**PC1**), followed by those more focused on relatively distinct, but comparable play styles—headshots (**PC2**), wounding (**PC3**, which implies some amount of cooperation), distance kills and sprees (**PC4**), and a facility towards teamplay (**PC5**).

To evaluate how these principal components are tied to group success, we sampled our merged dataset for groups of 4 friends who have played more than 30 games together, and ranked these groups according to their overall win ratio. A total of 28 groups meet this criteria. We then summed the principal component weights for all the group members to qualify the group's performance. Figure 3–10 depicts the top 10 groups in descending order: the best-performing group is placed at the left of the x axis. We observe that the three most successful groups have a positive summed PC weight for kills, assists and standing, and have a negative summed PC weight for headshots and kill distance. However, this trend does not visibly repeat itself for the remaining groups.

In Figure 3–11, we plot the summed PC weights of the bottom 10 groups in ascending order. Aside from the relatively low standings throughout the groups, there does not appear to be a discernable pattern to the PC weights which points to a lack of success. However, attention should be brought to the second-to-last group's summed PC weights. In this case, the signs of the principal components are all reversed with respect to the most successful top-10 group. This group's lack of



Figure 3–10: Summed PC Weights Versus Group Win Ratio (top 10). The merged dataset was sampled for groups of 4 friends who have played more than 30 games together. In an attempt to qualify each group's performance, we sum all the group member's PC weights, and plot them versus the group's win ratio. This plot provides the summed PC weights of the top 10 groups.



Figure 3–11: Summed PC Weights Versus Group Win Ratio (bottom 10). The merged dataset was sampled for groups of 4 friends who have played more than 30 games together. In an attempt to qualify each group's performance, we sum all the group member's PC weights, and plot them versus the group's win ratio. This plot provides the summed PC weights of the bottom 10 groups.

wins is likely explained by a low number of kills, very few headshots, a low amount of assists, a short distance between their opponents, and a low individual standing.

3.4 Discussion

Although not directly aimed at facilitating user analysis, vendor-sponsored game community sites provide a wealth of user data. Our use of the HALO: REACH Stats API has allowed us to quantitatively evaluate the cooperative behavior of FPS players in a competitive, team-based environment. The network structure of the sampled player set reveals an interesting correlation between winning and the number of online relations. It is not suprising that friends exhibit better team coordination, but our findings suggest that multiplayer game balance could be improved by matching competitive teams based on cohesion or some other group ranking metric. A further inspection of individual player behaviors in this multiplayer FPS context has yielded a set of five relatively stable descriptors tied to the underlying game mechanics of HALO: REACH; the intensely competitive setting tends to focus player types on success, which can be achieved either in general or through specializations.

Performance Metric	PC1	PC2	PC3	PC4	PC5
	(37%)	(14%)	(11%)	(9%)	(7%)
Games Played					
Score			-		+
Standing		-			(+)
Individual Standing		-			+
Kills	(+)				
Deaths		(-)		-	+
Assists	+			+	
Suicides					
Betrayals		-			-
Multi Medals	+	-			
Other Medals	+	-			
Spree Medals	+				
Style Medals	+			-	
Total Medals	+				
Unique Multi Medals	+	-			
Unique Other Medals	+			(+)	
Unique Spree Medals	+				
Unique Style Medals	+		-		
Total Unique Medals	+				
Avg Death Distance		+	(+)		+
Avg Kill Distance		+	+		+
Headshots	+	+		-	

Table 3–3: Round 1: simplified principal components. The first principal component (PC1) describes a player's ability at obtaining kills and medals, and is indicative of overall skill. PC2 contrasts a player's tendency to die with his/her capability at obtaining long-range headshots. For example, if a player has a large negative weight for PC2, this denotes that he/she is prone to death. Conversely, if the player's weight is large and positive for PC2, this denotes that he/she is apt at obtaining long-range headshots. PC3 identifies players whose deaths occur at a relatively long distance from their attacker's position. PC4 denotes a player's facility at obtaining Other medals, which correlates positively with assists. PC5 reveals how often a player loses; in the post-game results, a Standing value of 0 indicates a win, and a value greater or equal to 1 indicates a loss. As such, if a player's average Standing is a large value, this will be reflected as a large positive weight for PC5.

Performance Metric	PC1	PC2	PC3	PC4	PC5
	(39%)	(16%)	(10%)	(9%)	(6%)
Games Played					+
Score	+			-	-
Standing	-				-
Individual Standing	-				
Kills	(+)				
Deaths			(+)		(-)
Assists		+	-		
Suicides					
Betrayals			+	+	
Multi Medals	+		+		
Other Medals		+		+	
Spree Medals	+			+	
Style Medals	+	-		-	
Total Medals	+				
Unique Multi Medals	+		+		
Unique Other Medals		+			-
Unique Spree Medals	+				
Unique Style Medals	+			-	
Total Unique Medals	+				
Avg Death Distance		-	-	(+)	-
Avg Kill Distance		-	-	+	-
Headshots	+	(-)			

Table 3–4: Round 2: simplified principal components. PC1, which accounts for 39% of the variance in Round 2's player performance data, describes a player's ability at obtaining kills and medals. PC2 contrasts a player's ability at obtaining head shots versus his/her ability at obtaining assists; a negative weight along PC2 indicates a player's penchant towards accurate headshots, while a positive weight reveals his/her facility at obtaining assists. This duality is intuitive if we consider that a headshot is only attributed if it kills the opponent, while assists are only attributed to players who have helped defeat the opponent without delivering the final blow. Players with large positive weights for PC3 are prone to dying comparatively more often, while players with large negative weights appear to keep a larger distance between themselves and their opponents, and thus die less often, given the negative correlation. PC4 marks players who die at a longer distance from their opponents. PC5 correlates the average number of deaths per game with a player's tendency to obtain a high score. In objective-based games, many players sacrifice themselves in an attempt to secure the objective and collect more points.

Performance Metric	PC1	PC2	PC3	PC4	PC5
	(40%)	(15%)	(13%)	(7%)	(6%)
Games Played					
Score	+	-			
Standing					+
Individual Standing	-				
Kills	(+)				
Deaths		+		-	(+)
Assists			-	+	
Suicides				+	+
Betrayals		(+)			
Multi Medals	+	+			
Other Medals	+	+		+	
Spree Medals	+			+	
Style Medals	+			(-)	
Total Medals	+				
Unique Multi Medals	+	+			
Unique Other Medals	+		-	+	
Unique Spree Medals	+			+	
Unique Style Medals	+			-	
Total Unique Medals	+				
Avg Death Distance			(+)	+	+
Avg Kill Distance			+	+	
Headshots	+		+	-	

Table 3–5: Round 3: simplified principal components. PC1 identifies players whose versatility allows them to obtain more kills and more medals. PC2 interestingly defines a player's likelihood to betray his/her teammates. A plausible explanation is revealed if we observe that multi medals and unique multi medals correlate positively with betrayals in this component. Indeed, players who throw many grenades or use explosive weapons have a higher chance to accidentally kill their teammates in an explosion. These explosions nonetheless grant multi-medals if they successfully dispatch groups of opponents. As such, PC2 may allude to a player's preference at using explosives. PC3 contrasts long-distance and headshots with assists, similarly to the second principal component in Round 2. PC4 highlights a player's aptitude at obtaining style Medals, which is correlated with headshots (a headshot confers a headshot medal, which is categorized as a style medal). The positive value for assists naturally contrasts the negative value of headshots. Similarly to Round 1, PC5 correlates the average number of deaths and suicides per game with a player's likelihood to lose.
Performance Metric	PC1	PC2	PC3	PC4	PC5
	(39%)	(16%)	(10%)	(9%)	(6%)
Games Played					
Score	+	+	-		
Standing					(-)
Individual Standing	-				-
Kills	(+)				
Deaths		-			
Assists			-	-	
Suicides					
Betrayals		(-)			
Multi Medals	+	-			
Other Medals	+	-			
Spree Medals	+				
Style Medals	+			+	
Total Medals	+				
Unique Multi Medals	+	-			
Unique Other Medals	+			(-)	
Unique Spree Medals	+			-	
Unique Style Medals	+		-		
Total Unique Medals	+				
Avg Death Distance		+	(+)		
Avg Kill Distance		+	+		
Headshots	+	+		+	

Table 3–6: Round 4: simplified principal components. PC1 once again reveals the player's versatility at obtaining kills and medals. Players with a large negative weight along PC2 possibly prefer explosive weapons and grenades, similarly to PC2 in Round 3. This preference is contrasted with players who prefer long-distance, accurate headshots. PC3 denotes players who stay at a long distance from their opponents, and contrasts them with those who obtain many assists. Indeed, long-distance players may not have as many opportunities to do as much damage as assistive, short-range players. PC3 also contrasts assists with headshots, although the largest coefficient here pertains to other medals, attributed for melee kills and vehicle kills. Similarly to the previous PCA results, PC5 identifies players who are likely to lose more often.

Performance Metric	PC1	PC2	PC3	PC4	PC5
	(38%)	(14%)	(12%)	(9%)	(6%)
Games Played					
Score	+			+	
Standing					(-)
Individual Standing	-				-
Kills	(+)				
Deaths		+	-	+	-
Assists		+	(+)		
Suicides					
Betrayals		+	-		
Multi Medals	+		-		
Other Medals	+	+		-	
Spree Medals	+			-	
Style Medals	+	-		+	
Total Medals	+				
Unique Multi Medals	+		-		
Unique Other Medals		+	+	-	
Unique Spree Medals	+			-	
Unique Style Medals	+		+	+	
Total Unique Medals	+		+		
Avg Death Distance		-		-	
Avg Kill Distance		-		(-)	
Headshots	+	(-)			
	1				

 Table 3–7: Simplified principal components of the merged data set.

CHAPTER 4 Related Work

In this chapter, we present recent studies pertaining to the experiments described in this thesis. These works span a variety of fields including human-computer interaction, game studies, and behavioral psychology. Section 4.1 discusses recent efforts to design contemporary games using quantitative measurements, while Section 4.2 emphasizes works related to player behavior categorization.

4.1 Metric-Assisted Game Development and Adaptive Gameplay

Tracing its origins from the field of human-computer interaction research, the analysis of gameplay data has grown to occupy an important place in the game development industry, particularly within user testing frameworks. Under an academic light, efforts to explore the domain of game metrics have historically been restricted due to the propriatery nature of this data. However, only recently have new avenues been opened to explore parts of these large data sets, either through in-game or web-based API's. In this section, we provide a review of works related to gameplay metrics, and the domain's impact on game design and adaptive gameplay.

A variety of studies and efforts have been developed that aim to evaluate player performance, both during game design and as part of post-facto analysis. In prior work related to human computer interaction, for instance, the authors of [20] have developed the TRUE (Tracking Real-Time User Experience) system to collect streams of timestamped *user initiated events*, ranging in context from spreadsheet applications to video games. The TRUE system was used in the development of HALO 2 to isolate problem areas of the single-player campaign. A similar metric-oriented approach has been used in the development of HALO 3, namely to determine the fairness of multiplayer maps [31]. The TRUE system allows for a very detailed analysis of player behavior by combining video data, log file parsing, and user appreciation surveys.

The authors of [13] additionally attest to the benefit of game metrics in conjunction with the game design of FRAGILE ALLIANCE, and KANE & LYNCH, two modern first person shooters. The metrics collected during the testing phases of both these games consisted mainly of monitoring player health and positional data such as location and speed. In *Fragile Alliance*, the fairness of multiplayer maps was tuned by determining the location and cause of abnormal or un-intended distributions of player deaths on the map. The metrics collected during the testing phases of KANE & LYNCH allowed the level designers to tune the single-player encounters against the opponent AIs. Specifically, these encounters were balanced by observing the progression of the player's health according to his/her position on the map. The same authors have applied supervised AI learning techniques to metrics in TOMB RAIDER UNLIMITED to predict when a player will quit, or how long he/she will take to complete the game [23]. Of particular interest in this study was the observation that players who explored an underwater location at the beginning of the game were found to complete later puzzles more efficiently, owing to their exploratory nature.

The process of collecting gameplay metrics in order to complement game design is further illustrated in [12], where the authors describe the use of their own data collection system, TRACKTIVITY, during the testing phases of the racing game SPLIT/SECOND. Events triggered by testers were monitored by TRACKTIVITY to determine the difficulty of a race-track and to help designers tailor the game's overall learning curve. The difficulty of the computer opponents is also adaptively tuned to ensure the player is consistently challenged, without being overwhelmed. This tuning relies on the measurement of the player's post-race performance to determine an offset for the next race's AI difficulty value.

The success of adaptive difficulty mechanisms motivated by metric analysis has also been stated in a cooperative gameplay setting, namely in LEFT 4 DEAD [9]. This FPS features an AI DIRECTOR, which dynamically populates the map with zombie hordes. As cited in Section 2.2.2, a global *emotional intensity* metric is incremented when a player in the group is damaged. Additional arousing events, such as shooting a zombie at close-range, or being incapacitated, increment this emotional intensity value. Emotional intensity is decremented periodically when no arousing events occur. The AI director thus tunes the gameplay to insure an appropriate distribution of peaks and valleys in the temporal progression of the emotional intensity. This concept of emotional intensity serves as the basis for the WORLD OF WARCRAFT experiment presented in Chapter 2, which we supplement with the analysis of the group's position to measure the level of cooperation among its members. Adaptive difficulty in the MMO genre would notably increase player involvement, however the application of such a system in this genre remains to be implemented and studied.

4.2 Measuring Player Behavior and Player Types

The analysis of players actions within a game naturally leads to the categorization of observable player behaviors, or enduring player types. In this section, we present several studies which have attempted to classify player behaviors into intuitive categories, often tying into the domain of behavioral psychology to draw parallels between these categories and facets of human personality.

The seminal categorization of player behaviors according to specific types was first suggested by Bartle in [5]. In this work, the author initially classifies Multi-User Dungeon (MUD) players according to four basal axes: Achievers (Diamonds), Explorers (Spades), Socializers (Hearts), and Killers (Clubs). Achievers are generally motivated to obtain the best items in the game, either for a calculated advantage in a player-versus-player setting, or for prestige. Explorers typically seek knowledge of the game's mechanics, and are often the first to discover hidden areas, easter eggs, and glitches. Socializers leverage the community-building infrastructure of a game, for example: being a member of an organized guild, or focusing on item trading and the in-game economy. These players prefer interacting as a group, and may use the game to meet new people offline. Finally, Killers derive the majority of their enjoyment from player-versus-player competitive interactions and, occasionally, griefing (i.e. abusing game mechanics to prevent a player from progressing or otherwise playing the game as intended).

67

The MMORPG genre has indeed been a noteworthy vehicle of academic games research, most recently championed by quantitative assessments of player behavior in WORLD OF WARCRAFT [3, 21]. Recent work detailed in [37], for example, made use of survey data and behavioral metrics to identify the expression of psychology-based personality traits among players in WoW. Participants of this study completed a questionnaire to determine their alignment along the "Big-5" personality traits which include Extraversion, Agreeableness, Conscientiousness, Emotional Stability and Openness to Experience [15]. To measure the behavior of each participant a web crawler was developed to aggregate per-character achievement data in the WoW Armory over a period of 4 months. Personality traits were then shown to have significant correlations with various aspects of gameplay, suggesting that virtual worlds are an appropriate medium in which to infer personality traits. The sequential evolution of achievement-data mined from the WoW Armory has additionally been used to predict player behavior and character progression in [16].

Real-time player movement in WORLD OF WARCRAFT was investigated in a player-versus-player setting in [25]. In contrast to our method which polled the WoW client API for (x,y) player positional data, this study employed network packet inspection sent from the game server to the client to collect positional data of players in the environment. The data analyzed in this experiment isolated several important environmental hotspots which corresponded to gameplay objectives. Player movement patterns also suggested three distinct types of player behaviour: wanderers, patrollers and guards. Wanderers were distinguished by their erratic paths between ally and enemy-controlled hotspots, looking for battle. Patrollers only moved between ally-controlled hotspots, and concentrated on defending these areas. Guards were marked by minimal movement, and tended to stay around a hotspot of their choosing.

Relating to the social nature of multiplayer games, chat log analyses have been used in WORLD OF WARCRAFT to qualitatively survey group cooperation in 5-person instances [4]. The findings of this work coincide closely with the results of our WORLD OF WARCRAFT gameplay metric analysis. Namely, compelling dungeons provide a sufficient level of threat to engender social interactions among the players with the aim to coordinate, strategize, and gain control over the encounters.

Single-player first-person shooter games have also demonstrated archetypal player categories. The effect of level design on player behavior is investigated in [32], in which the authors use game metrics gathered from the stealth shooter game, HITMAN: BLOOD MONEY. Measurements including character position, rotation angle, actions performed and narrative choices are analyzed to qualify a player's overarching behavior, or *persona*. The game can thus label a player as either a *mass murderer*, a *silent assassin*, a *mad butcher* or *the cleaner*, depending on the prevailing nature of his or her actions.

The works cited in this section underline the relevance of studying game metrics as a vehicle to help develop engaging and balanced games. Previously bound to repeated testing and artistery, it has been shown that game design can be complemented by the quantitative analysis of gameplay data. The pertinence of such well-balanced games is evidenced by their longevity and by their large user bases. The continuous examination of gameplay data after a game is released is also an important factor in tuning game balance to avoid dominant — and therefore boring — strategies. These experiments have also illustrated that the analysis of game metrics is key to understanding player behavior and accordingly tailoring game content towards a broader audience. Our own work contributes to the advancement of this domain by extracting group-based metrics to evaluate cohesion and cooperation among players.

CHAPTER 5 Conclusions and Future Work

5.1 Conclusions

The objective of our work is to illustrate the use cooperative gameplay metrics as a toolset to isolate areas of games which require additional balancing. Proper balance is vital to the construction of a multiplayer game's community, as evidenced by the necessity for players to coordinate among each other to achieve their goals - be it to defeat an elusive dragon, or to take home the first place prize in a competitive tournament.

In Chapter 2 of our work, we measure cooperative behavior in a player-versus-environment setting of WORLD OF WARCRAFT. We make use of the WoW API to sample real-time gameplay data of several dungeon sessions, each of which being be matched to a specific bracket of character progression and equipment quality. Two metrics are employed in this chapter, namely: (1) player intensity and (2) the minimum enclosing circle. Player intensity measures the group's proportion of health lost per unit time, which reflects the difficulty of the scenario. The minimum enclosing circle determines the level of cohesion of a group; a tighter circle around a group indicates a heightened level of coordination and cooperation. Because the difficulty of a dungeon does not scale with respect to the group's stats (strength, health, power, etc), there is a notable drop in player intensity as players become equipped with better items. This is also mirrored by an overall drop in cohesion as illustrated by the larger minimum enclosing circle around the group; less stringent scenarios intuitively require less player attention and less coordination. These fine-grained cooperative metrics can therefore be used to evaluate the difficulty and required level of cohesion of a specific scenario's design. Our results here allude to the relevance of adaptively scaling the difficulty and pacing of a game to maintain player involvement.

By contrast, the procedure outlined in Chapter 3 measures cooperative gameplay from a different perspective. We focus here on HALO: REACH as a competitive, team-based, player-versus-player environment. The sheer volume of data provided by the HALO: REACH Stats API paints a broader picture of cooperative behavior. By crawling a sample of the total dataset we are able to build a graph of players who have been observed to play together in at least one game. We maintain edge weights as the number of times two players have played together in the same game. By analyzing the resulting network topology, and specifically by inspecting the progression of the average component size as edges are pruned by weight, we determine a weight threshold of 8 as the minimum number of times two players must play together before they can be considered as *friends*. With the ability to estimate friendships, we observe that players who play with their friends have a noticeably higher chance of winning than players who play alone. From one perspective, this may attest to a lack of balance within the underlying matchmaking system, and may suggest a benefit to exploring more elaborate group matching techniques. On the other hand, this result may be construed as a desired

property of the matchmaking system, which rewards the formation of close social groups as a means to promote the game's popularity and player retention.

In Chapter 3, we also applied an exploratory principal component analysis to the average player gameplay performances in an attempt to isolate specific player classes similar to the Bartle types laid out in Section 4.2. Our findings reveal a set of components whose weighted combination aptly describe a player's skill in relation to HALO: REACH's gameplay mechanics, specifically according to: (1) overall versatility, (2) ability to perform headshots, (3) ability to obtain assists, (4) aptitude at long-range combat, and (5) aptitude at cooperating with teammates. These results match our anecdotal expectation, specifically in light of the competitive and fast-paced nature of multiplayer first-person shooters. Based on the data collected, the resulting principal components do not eminently portray analogous Bartle types, however some rounds of data collection have alluded to the existence of explosive-weapon favoring players (betrayals, multi-kills), snipers (long-range, headshots), and close-range assault players (small-range, high assists).

5.2 Future Work

A wealth of future work is proposed in this section to build on the topics laid out in this thesis. Among the first directions to consider is the use of cooperative gameplay metrics in the analysis of TEAM FORTRESS 2 gameplay data. As a contemporary, popular, and free-to-play multiplayer first-person shooter, TEAM FORTRESS 2 innovates with a class-based approach to FPS team combat. Metrics such as team cohesion via the MEC and a time-series analysis of the class distribution per team would be an interesting mechanism by which to understand the dynamics of this game, and also to identify problem areas in map design and weapon/item balance. Both real-time data and post-game aggregate data are currently available for collection and analysis.

Additional cooperative metrics can be founded on a particular set of game mechanics for a specific game, however metrics based on group health or another resource system should be investigated in different genres to validate the generalizability of group intensity in regards to difficulty and pacing. Regardless of the game genre, an even more global metric to evaluate intensity or competitive tension would be the recording and analysis of audio communications among players in a group. In moments of very high intensity, one would expect players to speak quickly and loudly, while moments of exasperation would be marked by a dreary tone.

In a practical setting, the metrics developed in our work could serve as a basis for adaptive difficulty systems in cooperative player-versus-environment games. Specifically, the real-time computation of the MEC could be applied to different game contexts in which player proximity plays an important role. This would notably increase the replayability of otherwise static and predictable scenarios, and could be complimented by an equally adaptive reward system to benefit players who collaborate together, and thus reduce the appeal of freeloaders.

Lastly, further work in the domain of group ranking and matchmaking would hopefully yield more balanced win-to-loss ratios for tight-knit groups. Ideally, such groups would be matched against each other to ensure that lone-wolf players are also given the opportunity to play on an equal footing against one another. In such a hypothetically balanced system, community building incentives should nonetheless be implemented to promote socialization among these lone-wolf players.

References

- [1] Leigh Alexander. Gamers Hit 1.3 Billion Halo: Reach Matches. http://www.gamasutra.com/view/news/32111/Gamers_Hit_13_Billion_ Halo_Reach_Matches.php, December 2010.
- [2] Avery Alix. Beyond P-1: Who plays online? In Proceedings of DiGRA 2005 Conference: Changing Views-Worlds in Play, 2005.
- [3] Martin Ashton and Clark Verbrugge. Measuring cooperative gameplay pacing in World of Warcraft. In *FDG'11: Foundations of Digital Games Proceedings*, FDG '11, New York, NY, USA, 2011. ACM.
- [4] Shaowen Bardzell, Jeffrey Bardzell, Tyler Pace, and Kayce Reed. Blissfully productive: grouping and cooperation in world of warcraft instance runs. In *Proceedings of the 2008 ACM conference on Computer supported cooperative* work, CSCW '08, pages 357–360, New York, NY, USA, 2008. ACM.
- [5] Richard Bartle. *Designing Virtual Worlds*. New Riders, 2003.
- [6] Blizzard Entertainment. World of Warcraft subscriber base reaches 12 million worldwide. http: //us.blizzard.com/en-us/company/press/pressreleases.html?101007, October 2010.
- [7] Blizzard Entertainment. World of Warcraft. http://us.battle.net/wow/en/, December 2011.
- [8] Blizzard Entertainment. World of Warcraft API. http://www.wowwiki.com/World_of_Warcraft_API, February 2011.
- [9] Michael Booth. AI systems of Left 4 Dead. http://www.valvesoftware.com/ publications/2009/ai_systems_of_14d_mike_booth.pdf, 2009.
- [10] Bungie, Inc. bungiestats.jpg. http://www.bungie.net/images/News/Inline10/121710/bungiestats.jpg, December 2010.

- [11] Bungie, Inc. Halo Reach Online. http://www.bungie.net/stats/reach/online.aspx, December 2011.
- [12] Eduardo Jimenez Chapresto, Kenny Mitchell, and Francisco Jose Seron. Capture and Analysis of Racing-Gameplay Metrics. *IEE Software*, 28(5), 2011.
- [13] Anders Drachen and Alessandro Canossa. Towards gameplay analysis via gameplay metrics. In Proceedings of the 13th International MindTrek Conference: Everyday Life in the Ubiquitous Era, MindTrek '09, pages 202–209, New York, NY, USA, 2009. ACM.
- [14] Electronic Arts. Battlefield 3 Battlelog. http://battlelog.battlefield.com/bf3/servers/, December 2011.
- [15] L. R. Goldberg. The Structure of Phenotypic Personality Traits. American Psychologist, 48(1):26–34, 1993.
- [16] Brent Harrison and David L. Roberts. Using sequential observations to model and predict player behavior. In *FDG'11: Foundations of Digital Games Proceedings*, FDG '11, New York, NY, USA, 2011. ACM.
- [17] Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill(tm): A Bayesian skill rating system. In Advances in Neural Information Processing Systems 20, pages 569–576. MIT Press, 2007.
- [18] Roberto Ierusalimschy. Programming in Lua. Lua.org, 2nd edition, 2006.
- [19] I.T. Jolliffe. Principal Component Analysis. Springer-Verlag, 2nd edition, 2002.
- [20] Jun H. Kim, Daniel V. Gunn, Eric Schuh, Bruce Phillips, Randy J. Pagulayan, and Dennis Wixon. Tracking real-time user experience (true): a comprehensive instrumentation solution for complex systems. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 443–452, New York, NY, USA, 2008. ACM.
- [21] Chris Lewis and Noah Wardrip-Fruin. Mining game statistics from web services: a World of Warcraft armory case study. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, pages 100–107, New York, NY, USA, 2010. ACM.
- [22] LÖVE Development Team. LÖVE 2D. http://love2d.org/.

- [23] T. Mahlmann, A. Drachen, A. Canossa, J. Togelius, and G. N. Yannakakis. Predicting player behavior in Tomb Raider: Underworld. *Proc. IEEE Conference on Computational Intelligence and Games (CIG) 2010*, August 2010.
- [24] Microsoft Research. Trueskill(tm) Ranking System: Details. http: //research.microsoft.com/en-us/projects/trueskill/details.aspx, December 2011.
- [25] John L. Miller and Jon Crowcroft. Avatar movement in World of Warcraft battlegrounds. In *Proceedings of the 8th Annual Workshop on Network and Systems Support for Games*, NetGames '09, pages 1:1–1:6, Piscataway, NJ, USA, 2009. IEEE Press.
- [26] NetworkX Developers. NetworkX. http://networkx.lanl.gov/, November 2011.
- [27] Christian Nutt. A decade on, Halo charts its course. http://www.gamasutra. com/view/feature/6365/a_decade_on_halo_charts_its_course.php, May 2011.
- [28] Douglas A. Seifert. Minimum enclosing circle in Ruby. http://www.dseifert.net/code/mec/, February 2008.
- [29] Siqi Shen and Alexandru Iosup. The XFire Online Meta-Gaming Network: Observations and High-Level Analysis. In *IEEE HAVE 2011 Proceedings*, HAVE 2011, 2011.
- [30] Gillian Smith, Mike Treanor, Jim Whitehead, and Michael Mateas. Rhythm-based level generation for 2D platformers. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, FDG '09, pages 175–182, New York, NY, USA, 2009. ACM.
- [31] Clive Thompson. Halo 3: How Microsoft labs invented a new science of play. Wired Magazine, 09(15), 2007.
- [32] Anders Tychsen and Alessandro Canossa. Defining personas in games using metrics. In *Proceedings of the 2008 Conference on Future Play: Research*, *Play, Share*, Future Play '08, pages 73–80, New York, NY, USA, 2008. ACM.
- [33] Valve Corporation. Steam Community. https://steamcommunity.com/, December 2011.

- [34] James Whitehead and Rick Roe. World of Warcraft Programming: A Guide and Reference for Creating WoW Addons. Wiley Publishing, 2nd edition, 2010.
- [35] WoW Wiki. Item level. http://www.wowwiki.com/Item_level, August 2012.
- [36] Nick Yee. Motivations for play in online games. CyberPsychology and Behavior, (9):772–775, 2007.
- [37] Nick Yee, Nicolas Ducheneaut, Les Nelson, and Peter Likarish. Introverted elves & conscientious gnomes: the expression of personality in World of Warcraft. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 753–762, New York, NY, USA, 2011. ACM.