Achieving Balance in the ARPG Genre of Video Games with Asymmetrical Choices

Jeffrey J. Beyerl [*]

Abstract

Video games are a large and still growing industry. Games are becoming more complex and manufacturers are competing for players. An Action Adventure Roll Playing Game (ARPG) is one genre of game in which a player develops a character with a primary emphasis on combat encounters. This paper will give basic theory for describing and modeling balance in this type of game, with a particular look at the resource management scenario with asymmetrical builds.

1 Introduction

In an ARPG a player develops a character with his or her emphasis on efficiently winning player-versus-monster (PVM) combat encounters. In the first ARPGs there were very few, if any, choices. As computing technology and game design developed, players were given more choices. However, many of these choices are ineffective at stimulating player interest because there is one clearly "best" choice. The collection of all choices a player makes regarding the development of his or her character is called a build. Effective game balance will increase the longevity of the game by increasing the replayability. However, it should be noted that serious analysis in this field is still in its infancy [2].

One common method of giving players choices is through a resource management scenario. Typically this resource is called "mana", although the specific terminology is irrelevant and changes from game to game. Similarly a common term for a choice to be made is the selection of a character "skill". In this scenario the player has several choices for methods of creating a resource, and several choices for using this resource. This creates desirable complexity for the player: with just a linear number of choices, there are a quadratic number of builds. If there is one (or a few) builds that are significantly better than the rest, we say that such a build is dominant.

[*]Department of Mathematics, University of Central Arkansas, Conway, AR 7203 email: jbeyerl@uca.edu

We focus on combat encounters: both the player's character and his or her opponent(s) have a finite number of hit points. This a reasonable focus because the vast majority of time spent will be in combat. In fact, even in more social genres such as massively multiplayer online RPG's, players still spend over half their time in combat [3].

The ultimate goal in an encounter is to reduce the opponent(s) hit points to zero. Hence we quantify the damage per second (dps) with each resource generation skill and the dps with each skill requiring the expenditure of resources. Depending on the nature of the skills, the ratio of how often each can be used varies: we call the respective amounts uptime. The damage a character deals must be at least the hit points of the opponent, so we have the inequality

$$h \leq t \cdot (d_g \cdot u_g + d_c \cdot u_c)$$

where h is the opponents' hit points, t is the time taken in the encounter, $d_\bullet$ is the dps of each skill, and $u_\bullet$ is the uptime of the resource consumption skill in the specified build. There are more factors to address, as given in the next section, but this is a simple model of this scenario for one particular build chosen.

2 Modeling balance among builds

One goal of a game designer is to maximize the longevity of the game, which is ultimately achieved by depth and balance [1]. In an ARPG that means replayability: players should be encouraged to play the game many times, experimenting with each different build and not feel locked into just one. That is to say that all the different builds should be balanced: the time taken for a typical encounter should not be be significantly longer or shorter for one build than another. We can characterize the balance among builds by using statistical variance:

$$\frac{1}{n} \sum_b \left(\bar{t} - t_b\right)^2 \tag{1}$$

where t_b is the time taken in a typical encounter using build b, $\bar{t}$ is the average of all t_b, and n is the total number of builds. The goal, then, is to determine exactly how powerful each individual skill should be to nontrivially minimize the variance among builds.

In practice it can be desirable to specify how much time a typical encounter should take, so $\bar{t}$ will be treated as a constant. If the game involves nothing but simple combat (such as standing still and repeatedly swinging a sword until slaying the opponent), then the model in the previous section is enough. There are two more aspects of PVM combat that we consider here. Firstly a player will typically need to have his character do something other than attacking: blocking, dodging, kiting, feigning a retreat, or healing himself for instance. How much time he can spend actually dealing damage to the opponent will vary based on build, and we characterize that with the factor K representing the percentage of time spent on the offense. The letter K is chosen because players typically call this behavior “kiting”. Secondly, most games will have some skills that deal damage to a single target, and other skills that deal damage to multiple targets at once. We characterize this with the factor S for “splash damage”. Adding these to the model we have:

$$h \leq t_{g,c} \cdot K_{g,c} \cdot (d_g \cdot S_g \cdot u_g + d_c \cdot S_c \cdot u_c) \tag{2}$$

Every constraint forms a convex region. Because the feasible region of the problem is formed by intersecting each of these regions, it itself is convex. The objective is a convex function, and so there is a guaranteed optimal solution. The parameters depend on the game itself, but under this model there indeed is optimal balance for each game as stated in the following theorem:

Theorem 3 *Expression (1) with a specified $\bar{t}$ has a unique minimum subject to the constraints given by (2).*

3 A Simple Example

In this section we illustrate the simplest possible example that is not completely trivial. Assume that the combat is one player versus one opponent and that the player is able to attack continuously. The player will have two options for resource generating skills, and three options for resource consuming skills. Denote these as g_1, g_2, c_1, c_2, c_3. There are then six possible builds, each with a possibly different length of engagement: $t_{g_1,c_1}, t_{g_1,c_2}, t_{g_1,c_3}, t_{g_2,c_1}, t_{g_2,c_2}, t_{g_2,c_3}$. Denote the hit points of the opponent as h, and the uptime of the resource consuming skill in each build as $u_{g_1,c_1}, u_{g_1,c_2}, u_{g_1,c_3}, u_{g_2,c_1}, u_{g_2,c_2}, u_{g_2,c_3}$. We then have the conditions given below.

$$\begin{aligned}
h &\leq t_{g_1,c_1}\left(d_{c_1} u_{g_1,c_1} + d_{g_1}(1 - u_{g_1,c_1})\right)\\
h &\leq t_{g_1,c_2}\left(d_{c_2} u_{g_1,c_2} + d_{g_1}(1 - u_{g_1,c_2})\right)\\
h &\leq t_{g_1,c_3}\left(d_{c_3} u_{g_1,c_3} + d_{g_1}(1 - u_{g_1,c_3})\right)\\
h &\leq t_{g_2,c_1}\left(d_{c_1} u_{g_2,c_1} + d_{g_2}(1 - u_{g_2,c_1})\right)\\
h &\leq t_{g_2,c_2}\left(d_{c_2} u_{g_2,c_2} + d_{g_2}(1 - u_{g_2,c_2})\right)\\
h &\leq t_{g_2,c_3}\left(d_{c_3} u_{g_2,c_3} + d_{g_2}(1 - u_{g_2,c_3})\right)
\end{aligned}$$

Here the parameters are the values $u_{i,j}$ and h. The set of feasiable solutions are all combinations of $t_{i,j}$ and d_i that satisfy all of the above. We are particularly interested in the five values $d_{g_1}, d_{g_2}, d_{c_1}, d_{c_2}, d_{c_3}$ because these are what will be used to balance the game.

For the purpose of an example, we'll choose parameters as given below.

Resources				
g_1	g_2	c_1	c_2	c_3
5	10	-10	-20	-100

Uptimes			
$u_{i,j}$	c_1	c_2	c_3
g_1	$1/3$	$1/5$	$1/21$
g_2	$1/2$	$1/3$	$1/11$

Above the Uptimes are computed from the resources generated and consumed. For the purpose of illustration we specify 6000 hit points in which case we arrive an the following optimal solution in which every build completes the battle in 30 seconds with the damage values below.

Optimal Damage				
g_1	g_2	c_1	c_2	c_3
175	150	250	300	700

4 A Real Example

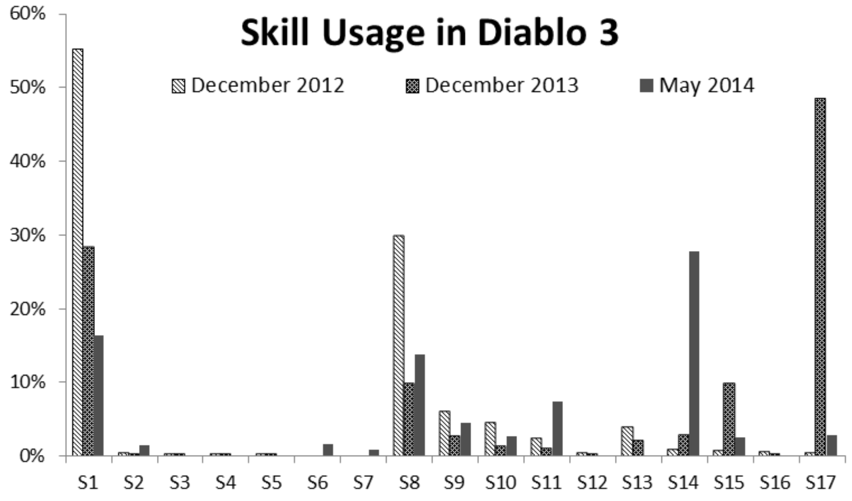
For an example that is more realistic, we turn to the 2012 ARPG Diablo 3. In this game a player chooses 6 skills to use, from a pool of approximately one hundred. Typically one of these would be a resource generator, one a resource consumer, and four utility skills. The the two years following release the developer (Blizzard Entertainment) continued the balancing process through 6 substantive patches.

While some utility skills require or generate resources, we focus here on the primary source of generation and consumption. The Witch Doctor character class in particular has 19 skills that can be considered generators, and 15 skills that can be considered consumers. Denote each of these sets as $\mathcal{G}$ and $\mathcal{C}$. Hence $|\mathcal{G}| = 19$ and $|\mathcal{C}| = 15$. This results in 285 constraints and an objective with 34 terms.

$$\text{Minimize} \sum_{b \in \mathcal{G} \cup \mathcal{C}} \left(\bar{t} - t_b\right)^2$$

$$\text{Subject to } h \leq t_{g,c} \cdot K_{g,c} \cdot (d_g \cdot S_g \cdot u_g + d_c \cdot S_c \cdot u_c) \ \forall (g,c) \in \mathcal{G} \times \mathcal{C}$$

This is a particularly nice example for academic studies because every player's profile is available online. The evolution of game balance can be illustrated in the figure below that gives each player's selection of resource consumption skills at three selected timestamps. The vertical axis is the percentage of characters using each particular skill. Note that these do not add up to 100% because players are allowed to choose multiple or none of these skills (although it is typically a poor decision to do so). Players can freely change their skills, so it is reasonable to expect that the skills a player chooses are the skills he or she prefers to use.



Code	Skill	Code	Skill
S1	Zombie Bears	S10	Lob Blob Bomb
S2	Explosive Beast	S11	Corpse Bomb
S3	Leperous Zombies*	S12	Kiss of Death
S4	Wave of Zombies*	S13	Dire Bats
S5	Undeath	S14	Vampire Bats*
S6	Pile On*	S15	Plague Bats
S7	Lumbering Cold*	S16	Hungry Bats
S8	Acid Rain	S17	Cloud of Bats
S9	Slow Burn		

*Some skills were introduced, removed, or redesigned

Notice in particular that in 2012 the vast majority of players used just two skills, S1 and S8. The fact that the graph is closer to uniform in 2014 shows that the changes to the game improved balance. It also illustrates just how difficult it is to balance a game: for instance less than a tenth of a percentage of the population choose to use some skills such as S12 and S16.

As a specific example of the balancing process, let us consider skill S17. It is a low-range attack with that increases in damage while holding your position in combat, such a dynamic is called triangularity[4]. A model such as the one presented in this paper gives a starting point, but ultimately a designer should expect to perform corrections as they discover how players feel about the game [4]. In the original version of the game this skill dealt 334% of a player's weapon damage, which was low enough that the vast majority of players considered it to be completely unusable. Then in an attempt to encourage usage, its damage was increased to 600% where it became nearly dominant. In the third correction its damage was reduced to 525% where it is no longer dominant, but this was an overcompensation to the point that only about 4% of the player base continued to use it.

5 Conclusions

Game balance is an important but often elusive goal in the development of video games. In the ARPG genre one aspect of balance is in the diversity of build selection that players can choose from. Players will ultimately have to make choices that impact the effectiveness of their character and the enjoyment they obtain from the game. In a balanced game these will not be opposing choices: players would not feel forced into a single dominant build. We characterized the effectiveness of a build in terms of the efficiency in time taken during combat encounters. The model given in section 2 uses the resource generation and consumption scenario and further takes into account the gameplay mechanics of splash damage and kiting. The parameters of the model are based on the mechanics of each individual skill, the damage assigned to each skill can then be determined to minimize differences in effectiveness between builds.

References

[1] Dave Morris Andrew Rollings. *Game Architecture and Design.* New Riders, Indianapolis, Indiana, 2003.

[2] Jesper Juul. *Casual Revolution.* The MIT Press, Cambridge, Massachusetts, 2010.

[3] Maja Matijasevic Mirko Suznjevic, Ognjen Dobrijevic. Hack, slash, and chat: A study of players' behavior and communication in mmorpgs. *Network and Systems Support for Games (NetGames)*, 2008.

[4] Jesse Schell. *The Art of Game Design.* Elsevier, Burlington, 2010.