When Losing Means Winning: Algorithmic Fairness in Mario Kart

Audrey Shin, Felix Holmes, Rachel Claire Henry

May 7, 2025

1 Introduction

The release of the multiplayer game *Super Mario Kart* in 1992 revolutionized the genre of racing games. Since inception, the Mario Kart series has sold over 188.92 million copies with a worldwide player base spanning across Asia, Europe, America and other regions. This paper focuses primarily on the algorithms used in *Mario Kart Wii* (2008), which sold more than 37 million copies and remains one of the most commercially successful Wii games, second to *Wii Sports* [8, 3]. In *Mario Kart Wii*, players compete against up to twelve players, which can include both humans and AI controlled racers (CPUs) in themed courses ranging in difficulty with the goal of finishing in first place. Unlike conventional racing games, which reward consistent driving performance, the Mario Kart series differentiates itself with its utilization of several algorithmic balancing methods to ensure no race is truly ever over until everyone crosses the finish line, so both experts and novices can play together and have fun.

At key points during each race, players can drive through glowing item boxes that assign players a power up. These items range in utility where items like a Banana or a Blooper offer minimal advantage, and 'S-tier' items like a Bullet Bill or a Star can boost a poorly performing player to first place. There is online discourse about the exact ranking of the items; however, there are generally items that are irrefutably more powerful than others.

Figure 1: Unofficial Mario Kart Wii item tier list created by a Reddit user. Taken from a popular Reddit post with 224 upvotes [5].

The current literature suggests that the likelihood of receiving certain items is based on a position-dependent weighted discrete probability distribution[9]. This implies that players who are worse off will be rewarded with better items to keep them in the race. This system, known as item probability scaling, is one of the several techniques Nintendo uses to maintain competitiveness between the players. In addition to the item

distribution algorithm, *Mario Kart Wii* employs adaptive behavior in CPUs that become more aggressive when trailing or seem to slow down when they have too much of a lead. Earlier games in the series like *Mario Kart 64* (1996), have obvious, scripted rubber banding AI where rival CPUs are hardcoded to remain unnaturally near other players. Rubber banding brings players stretched across various distances closer by snapping them together using a variety of algorithms. This means that CPU racers can unrealistically speed up to players that had a large lead even if it should be impossible. These mechanics all fall under the broader concept of Dynamic Difficulty Adjustment (DDA), a philosophy that alters the game difficulty based on the player's performance in real time. Though more recent titles in the series like *Mario Kart Wii* rely less on scripted CPU rubber banding, item distribution algorithms are still heavily employed.

These systems raise the question of algorithmic fairness. In particular, whether Mario Kart's DDA technique is algorithmically fair, or if it disadvantages individual high performing players in the interest of group fun. While DDA aims to keep the game fun for everyone, it penalizes skill to make the game casual friendly. This paper investigates the balancing systems used in *Mario Kart Wii* from an algorithmic perspective looking at the tension between individual skill and group enjoyment, and further explores the implications of fairness in algorithmic design.

2 System Description

Due to the proprietary nature of Nintendo's game design, the exact implementation details of the DDA technique used in *Mario Kart Wii* remain undisclosed. However, based on observed game play and design conventions seen across the racing game genre, the consensus is that *Mario Kart Wii* employs a combination of two well known player balancing techniques: item based probabilistic balancing and rubber banding. These techniques are designed to reduce player performance disparities by adjusting in-game variables such as item boxes or game physics based on the position of the player in a race, thereby introducing a dynamic element of fun and unpredictability.

Item based probabilistic balancing manipulates the distribution of item boxes. Specifically, players in lower positions are more likely to receive more advantageous items such as Bullet Bills, whereas players in higher positions are more likely to receive less advantageous items such as a Banana. In this problem, we assume the item probability algorithm uses the following inputs: player position, lap number, historical performance, and items. The current player position represented by p_i is a number from 1 through 12 (the amount of players allowed in each race). The lap number is l_i, and the set I is the set of available items Mushroom, Triple Mushroom, Golden Mushroom, Mega Mushroom, Green Shell, Triple Green Shells, Red Shell, Triple Red Shells, Spiny Shell, Banana, Triple Bananas, Bob-omb, Fake Item Box, Bullet Bill, Star, Blooper, POW Block, Thunder Cloud, Lightning. The output is a weighted probability distribution over the items where more powerful items have higher weights when p_i is higher. Let $W_i = w_1, w_2, \ldots$, be the weights of each item based on the position.

Local VS Races, drivers controlled by players												
Item	Rank											
	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th
	65	35	25	15	5							
		50	45	40	25	15	10					
	75	35	20	5	5							
	35	20	10	5								
		20	35	45	35	30	20	15	5			
			5	15	30	45	55	65	75	65	45	15
		5	10	15	10	10						
				5	10	15	15	10	5			
												45
							10	25	35	50	45	35
						5	20	40	50	70	70	40
				5	15	20	20	10	10			
					10	10	10	10	10			
					10	10	15	10	10			
			10	15	15	15	10	5				
										15	40	65
		10	20	20	10	5						
			5	10	20	20	15	10				
	25	25	15	5								

Figure 2: Item probability distribution for local VS races in *Mario Kart Wii*. Reproduced from Super Mario Wiki [7].

However, rubber banding alters game physics or CPU behavior to narrow down gaps between players. That is, CPUs may speed up unnaturally. It is assumed that the CPU rubber banding adjustment algorithm uses a position gap δ p between the CPU and the leading player, and the current speed of the CPU s_i. The output would be the adjusted speed s_i' of the CPU where $s_i' > s_i$ if trailing and the opposite if leading. We also discuss an alternative solution to unfair player match ups in Section 3.3.

3 Alternative Algorithms

3.1 Player Independent Item Distribution

Though there are more subtle CPU rubber banding algorithms used in *Mario Kart Wii* than in *Mario Kart 64*, position dependent item distribution probabilities give lower ranked players advantages that can drastically skew results. This solution proposes a position independent weighted distribution where all players receive items based on the same fixed probability distribution. In other words, more powerful items will be more rare, but the rarity is agnostic of the position of the player in the race. This algorithm's focus is to reward individual players and de-penalize higher skilled players. As stated in the problem statement, $I = i_1, i_2 \ldots i_k$ is the set of all possible items in the item boxes. Instead of using a set, because the probabilities of getting a certain item are fixed, a dictionary can be used. For all players the item received is drawn from $D = i_1 : w_1, i_2 : w_2 \ldots i_k : w_k$, the dictionary containing the items and their corresponding probabilities of appearing in the item box. The actual fixed probabilities should be chosen such that powerful items are

more rare. Here the item rarity is global and position independent.

A simple implementation of the algorithm in the game could be a quick three second weighted roulette wheel that pops up on the side, where less powerful items would take up a higher fraction of the wheel corresponding to their higher probability of being chosen, and the wheel would be the same for all players regardless of their position. After the player drives through the item box, assuming the cumulative distribution is already precomputed, generating a random number takes constant time $O(1)$. Given that the for loop goes through at most m items in the list, the worst case running time is $O(m)$. However, since m is a relatively insignificant constant number ($m < 19$), it is will be $O(1)$ in practice because as the game grows larger this number remains constant.

Algorithm 1: Fixed Item Distribution Algorithm

Input: Item probability dictionary $D = \{i_1 : w_1, i_2 : w_2, \ldots, i_k : w_k\}$
Output: An item randomly drawn based on the fixed distribution

```
random_num ← random number between 0 and 1
cumulative ← 0
foreach (item, weight) in D do
    cumulative ← cumulative + weight
    if random_num ≤ cumulative then
        return item
    end
end
```

Though this algorithm intends to reward individual skill, it may inadvertently reduce it by limiting opportunities for comebacks, decreasing competitiveness. Additionally, because the weights are fixed, this can make catching up to those in the lead impossible if they happen to get a 'S-tier' item in the last lap, for example.

3.2 Circuit Adaptive Track

While a global item distribution aims to prioritize individual fairness, a circuit adaptive track emphasizes group fairness, acting as a possibly more equitable alternative technique to rubber banding. Instead of getting a boost from a Bullet Bill from 12th to 2nd place, a player in last place might receive an adapted version of the track to race on in real time, in order to catch up with players who are ahead. The paper *Circuit-adaptive challenge balancing in racing games* by Rietveld et al., provides an algorithm for this approach [1]. The track is sectioned up into several parts, with each part having different versions. Harder versions will have narrower roads and sharper turns, while easier versions will have both wider roads and turns.

Algorithm 2: Circuit-adaptive rubber banding

```
1  while race-not-finished do
2      for all-players do
3          if player-completes-circuit-segment and next-circuit-segment-unoccupied then
4              if gap-size-within-designer-specified-tolerance then
5                  c ← SVMClassifier(gameplay-observations)
6                  switch c do
7                      case too-easy do
8                          inject-hard-circuit-segment
9                      end
10                     case just-right do
11                         inject-circuit-segment-with-same-challenge-level-as-current
12                     end
13                     case too-hard do
14                         inject-easy-circuit-segment
15                     end
16                 end
17             end
18             else
19                 if player-is-too-far-behind then
20                     inject-easy-circuit-segment
21                 end
22                 else
23                     inject-hard-circuit-segment
24                 end
25             end
26         end
27     end
28 end
```

Adapted directly from Rietveld et al. (2014) [1].

The general outline for this algorithm detailed in the paper by Rietveld et al. is a while loop that continues until the race is completed. Within this while loop the algorithm will iteratively check each player's rank within the race and adjust the circuit accordingly. The algorithm will not change the difficulty of a segment if there is a player currently on that segment. The way Rietveld et al. suggests checking and adapting the circuit is using a designer specified tolerance level of how far behind a player is from the one in front of them and a switch case to determine which difficulty to change the circuit to. The switch case allows for more specific instances and other gameplay data to be considered, but the authors also include the option to simply rubber band in the binary sense of if someone "is too far behind" or not.

This algorithm uses rubber banding and applies it to help players who are behind, catch up by giving them an easier. Simultaneously the algorithm slows down players in higher ranks by giving them more challenging paths. This achieves the goal of lessening the margin between those who are ahead and far behind, making all players more balanced.

The authors consider altering the speeds of players depending on how they are performing as traditional rubber banding. They argue that circuit adaptive rubber banding is simply an alternative method for player balancing. They also suggest that this method of rubber banding can coincide with traditional rubber banding.

This algorithm depends on two variables. One variable being duration d of the race and the other being the number of players within the race n. The while loop depends on how quickly players complete the race, and the inner for loop depends on the number of players within the race, which is bound to twelve players. These two loops are nested, thus the overall complexity of the algorithm is approximately $O(d)$.

3.3 Matching players with similar profiles/skill levels

Rather than adapting to the game as it is being played, this algorithm attempts to player balance before the race is in progress through matching up players with similar abilities. This approach would also thus prioritize group fairness over individual fairness through attempting to level the playing field. An algorithm that can be used to match players is a merge sort algorithm.

In *Mario Kart Wii*, there are different types of tracks or race courses, to which number IDs will be assigned. While a player profile may be more generalized, a player profile can be taken as a ranking of the race courses in ascending order of how well a player has performed in each course. Although there might already be players grouped together based on average times or other data points, this paper will focus on course rankings for the purposes of this example.

Without loss of generality, compare two players A and B. Let player A's profile ranking be the "sorted" order of the tracks. Apply Merge and Count to player B's profile ranking, and then sort them according to the "sorted" order from player A. By counting the number of inversions, it will become clear how similar or dissimilar a pair of players are. That is, with more inversions, the more dissimilar the pair of players, whereas with less inversions, the more similar the pair of players. Again, this algorithm is relatively intuitive. The goal is to match players with similar players so that there is an even matching of skill level. When players have similar abilities, there will be more balance within the game, allowing more competitiveness.

In a paper by Cechanowicz et al., they suggest using several different balancing methods to increase overall group enjoyment. Furthermore, they show that using multiple player balancing techniques, in fact, leads to more effective player balancing and is less noticeable to players [2]. Thus, this approach in conjunction with other approaches, an create a more well rounded player balance within the game. Additionally, because this method would be used before the game is played, it can be effectively used in addition to the techniques that would occur during game play.

This algorithm will run in $O(nlogn)$ when comparing two players. n in this scenario refers to the number of tracks, which is a constant number. Thus, the actual complexity of this algorithm is constant time. However, the real-time complexity extends beyond the scope of the comparing algorithm. It is particularly significant in how it is applied to a group of players. For example, if there are n players, and each player is compared with every other player, the time complexity will approximately be $O(n^2)$.

4 Fairness Analysis

The tension of fairness that *Mario Kart Wii*'s design highlights is mainly that of individual fairness and group fairness. As simply put by Friedler et. al, individual fairness aims to treat individuals who share similar features in the same way [4]. Thus, in the context of game play, the standard approach of individual fairness holds that outcomes such as overall victory or item distribution should reflect a player's in-game skill and performance: a player who executes better racing lines and smarter strategies should be rewarded with a higher likelihood of winning, whereas a player who does not should lose. In contrast, group fairness takes the approach of non-discrimination, where systems are adjusted to people's backgrounds. *Mario Kart Wii*, in particular, implements this through DDA mechanisms like item-based probabilistic balancing and rubber banding. As previously mentioned, these techniques appear to narrow the performance gap between players by distributing more advantageous items to those in lower positions in the race and less advantageous items to those in higher positions as well as manipulating the speed of CPUs depending on real players' game play. Thus, *Mario Kart Wii* appears to hold the group fairness idea where everyone should have an equal opportunity to win regardless of skill level, which it achieves through maintaining competitive balance among all players.

While the implementation of these mechanisms certainly make the game more fun and unpredictable for its audience, they raise important questions regarding the issue of algorithmic fairness. That is, should players be inadvertently penalized for performing well? It is also important to consider a key third component in this discussion: perceived fairness. Even if the *Mario Kart Wii* system aims to balance player outcomes objectively, it can still be perceived as unfair to the game's players themselves. Less skilled players may perceive this balancing as welcoming, since it helps them stand a chance against better competitors. However, these accommodations might result reduced enjoyment or engagement for more highly skilled players, who do not benefit from these accommodations.

The fairness discussion also reflects larger debates surrounding algorithmic fairness. For example, Professor Reuben Binns discusses both individual fairness and group fairness as it relates to college admissions and financial lending. More specifically, he shows how either approach might be used to mitigate disparities depending on how an individual's performance is perceived. That is, whether the performance disparity appears to be due to an individual's effort and choice or structural disadvantage [6]. Now applying this to the "real world", in the case of college admissions, affirmative action can be seen as rubber banding where outcomes are adjusted based on "players" who may have greater systemic barriers. Thus, like in *Mario Kart Wii*, this technique aims to create a more level playing field, foster inclusivity and ensure that colleges are accepting diverse communities. Additionally, Binns' brings up the example of financial lending and how women may be seen as less financially credible compared to men despite similar financial histories. The question, then, is whether this is the result of personal circumstances or a larger disadvantage that needs to be dynamically adjusted for. In *Mario Kart Wii*, if victory were to be solely based off of skill, artificial interference in game play would be eliminated, which includes removing rubber banding. This would be analogous to only considering personal circumstances in the case of college admissions and financial lending. In contrast, if a more holistic approach were desirable, then low-skilled players would continue to be accommodated through *Mario Kart Wii*'s current DDA mechanisms. This would be analogous to taking into consideration larger societal marginalization of certain groups.

However, it is important to note that the question of whether or not something is definitively fair is difficult to answer, and largely depends on the system's goals. The goals of *Mario Kart Wii*, for example, might simply be to be fun, inclusive, and enjoyable for all, which might require a different criteria of fairness than that of other games or real life situations.

5 Conclusion

Mario Kart Wii acts as an interesting case study regarding algorithmic fairness, especially through its DDA mechanisms such as item probability scaling and rubber banding. While designed to maximize engagement and enjoyment amongst players, these mechanisms also bring to light important questions regarding fairness for highly skilled players. In our paper, we have mainly focused on the tension between individual fairness, which rewards individual skill, and group fairness, which prioritizes group inclusivity. We have also proposed alternatives to the DDA mechanism approach to fairness present in *Mario Kart Wii* such as random item distribution, circuit adapting, and skill matchmaking, which attempt to minimize disparities between high-skilled players and low-skilled players in different ways. Importantly, we note that there is no singular solution to fairness, and, in fact, it depends on a system's goal and context. In the case of *Mario Kart Wii*, while the mismatch between competition and enjoyment are intentional, our algorithmic analysis allows us to better understand, and possibly improve, designs in gaming and beyond.

References

[1] A. Rietveld et al., *Circuit-adaptive challenge balancing in racing games*, 2014 IEEE Games Media Entertainment (GEM), Toronto, ON, Canada, 2014, pp. 1–8. doi: 10.1109/GEM.2014.7048075.

[2] Cechanowicz et al., *Improving Player Balancing in Racing Games*, University of Saskatchewan, Saskatoon, Canada, 2014, pp 47-56. doi: 10.1145/2658537.2658701

[3] ContentEngine Noticias Financieras. *What is the history of Nintendo's Mario Kart video game? Gale OneFile: News*, 26 Mar. 2025. Available at: https://link.gale.com/apps/doc/A832934878/STND?u=mlin_m_wellcol&sid=ebsco&xid=7eeac09a.

[4] Friedler et al. (2021) *The (Im)possibility of Fairness: Different Value Systems Require Different Mechanisms For Fair Decision Making*, Communications of the ACM. doi: https://doi.org/10.1145/3433949

[5] Reddit user u/[Gz0njh]. *This is probably the factual item tier list for Mario Kart Wii.* Reddit. Available at: https://www.reddit.com/r/MarioKartWii/comments/ennfi9/this_is_probably_the_factual_item_tier_list_for/.

[6] Reuben Binns (2019) *On the Apparent Conflict Between Individual and Group Fairness*, Communications of the ACM. doi: search.ebscohost.com/login.aspx?direct=true&AuthType=ip,sso&db=edsarx&AN=edsarx.1912.06883&site=eds-l

[7] Super Mario Wiki contributors. *Mario Kart Wii item probability distributions – Local VS Races, drivers controlled by players.* Super Mario Wiki. Available at: https://www.mariowiki.com/Mario_Kart_Wii_item_probability_distributions.

[8] VGChartz Wiki. *Mario Kart Series Sales Figures.* VG Sales Fandom Wiki. https://vgsales.fandom.com/wiki/Mario_Kart.

[9] Q. Mi and T. Gao, *Adaptive rubber-banding system of dynamic difficulty adjustment in racing games*, ICGA Journal, vol. 44, no. 1, pp. 18–38, 2022. doi: 10.3233/ICG-220207.