Design notes: A completionist's nightmare

Charles Zheng

January 5, 2014

1 Introduction

A completionist's nightmare is a single-player card-collecting game which procedurally generates infinitely many cards to collect, and which uses performance-based bonuses to incentivize optimization and careful play as opposed to mindless grinding. By taking advantage of asymptotic probabilistic phenomenon and using simulations, we can control the scaling of the game as gameplay time goes to infinity.

2 Card collection scheme

There are infinitely many card sets; players receive random cards as rewards for a particular challenge. There are $j = 1, \ldots$ levels of challenges with 2^j challenges each. Each level of challenges shares the same card set. However, to reward the unlocking of other challenges on the same tier, each challenge per tier also has a unique card set. The player participates in the challenge and wins or loses. Based on their win or loss record, we measure a performance rating. After attempting a challenge, the number of randomly selected cards they receive is roughly exponential in their performance.

If $X_1, \ldots, X_n$ is the player's win-loss record for the challenge, the number of cards they receive from the shared card set is

$$1 + X_n * \max_{0 \le i \le j \le n} K_j^{\frac{\sum_{k=i}^{j} X_i}{j-i+1}}$$

and

$$X_n * \max_{0 \le i \le j \le n} \left(\frac{K_j}{2^j}\right)^{\frac{\sum_{k=i}^{j} X_i}{j-i+1}}$$

from the challenge-specific card set, where K_j is determined based on game complexity and game difficulty scaling considerations. The rationale for the bonus system is described in my notes on 'rating systems' for single-player games.

The distribution of cards in each card set is as follows. There are 2^i cards in each rarity tier $i = 1, 2, \ldots$, occurring with probability $(1/4)^i$) In each tier the player has collected 2^{2k-i} out of 2^i cards, thus the fraction of cards in that tier which they have collected is $(1 - e^{-2^{2(k-i)}})$. This means that if the player has collected $n = 2^{2k}$ cards, then the number of unique cards they own is

$$\sum_{i=1}^{\infty} (\text{no. of cards in tier } i)(\text{percent obtained}) \approx \sum_{i=1}^{\infty} 2^i (1 - e^{-2^{2(k-i)}}) \approx 2.55\sqrt{n}$$

Furthermore, two different players who have each collected the same number of cards from each set will have about 58% of their collection in common, since

$$\sum_{i=1}^{\infty} (\text{no. of cards in tier } i)(\text{percent obtained})^2 \approx \sum_{i=1}^{\infty} 2^i (1 - e^{-2^{2(k-i)}})^2 \approx 1.49\sqrt{n}$$

This is ignoring card sets unique to challenges. Suppose the player has unlocked all challenges on a level and has the same performance on each of them, has a collection multiplier of B for the main set and thus B for the challenges. Then if the number of wins gained by the player is w, the number of wins for each challenge is $2^{-j}w$, and the player owns $2.55\sqrt{Bw}$ unique cards from the main set and $2.55\sqrt{2^{-j}B2^{-j}w}$ from each challenge, hence $(2.55)2^j\sqrt{2^{-j}B2^{-j}w} = 2.55\sqrt{Bw}$ from all challenges combined, thus doubling their collection rate compared to a player who has only unlocked one challenge.

Recall that the bonus factor can reach up to 10000. Thus, better skills can translate into 100 times as many cards at the same "stage" of the game.

3 The card game

In a duel, each combatant has a deck of at least 10 cards (no repeats) and draws a hand of 5 cards. In each round, both combatants simultaneously select a card from their hand; the cards are revealed, and one card is eliminated while the other returns to the owner's hand. The combatant whose cards are eliminated loses the duel.

The combat mechanic is basically an extension of rock-paper-scissors. First of all, instead of three elements (rock, paper, scissors), there are five. With three elements, the optimal decks will be generally be balanced, which leads to poor variety. We can restrict decks to have at most two elements, but then the actual game play suffers, since a player has only two different kinds of choices in each round of combat. But with five elements, we can restrict each deck to have three elements, so that 10 different elemental combinations are possible, and players can have up to 3 different choices in combat.

The elements are Green (nature), White (law), Red (chaos), Blue (knowledge), and Black (death), abbreviated GWRUB. One might recognize the color "wheel" from Magic: the Gathering, but unlike Magic, we explicitly specify which colors dominate which other colors.

- Green beats White and Red, since the mighty beasts of nature overpower organized armies and unruly mobs alike.
- White beats Red and Blue, since disciplined peacekeepers subdue troublemakers, while research depends on the resources of an orderly society
- Red beats Blue and Black, since chaos disrupts the controlled conditions needed for seekers of knowledge, while anger can dominate over the fear of death
- Blue beats Black and Green: science provides the power to triumph over death and tame nature
- Black beats Green and White: death ultimately consumes all living beings, no matter how great; death cares nothing for titles and authority

Whether or not a particular card beats any other particular card depends on their elements: if both cards have the same element, a coin flip determines the winning card. But there are exceptions. Each card can beat a few cards of the same type or of the type it is normally weak against. Thus the set of attributes for any card are:

- Card set and number
- Color
- Card name

- Counters:
- Is countered by:

4 Complexity growth

The amount of relevant information for each card are its type, the cards it counters, and the cards it is countered by. We wish to control the growth rate of the total amount of information the player has to know at the jth level of challenges, by having

$$\text{pieces of information} = O(j \log j)$$

The number of cards collected from each of the $i = 1, \ldots, j$ tiers is $O(K_i)$. In addition to type, the player needs to remember a number of pairs (i, j) where card i counters card j. We generate card attributes by the following: a card in card set for level t has a probability $p_{t \vee s}$ of countering or being countered by a card in card set s. In the section on challenges we consider setting p_t to control the difficulty growth of the game. The number of cards per set is controlled by $\sqrt{K_j}$. Given the information growth rate we determine K_j by

$$j \log j \approx \sum_{i=1}^{j} 1 + \log(i) \propto (p_1 (K_1)^2 + p_2 (K_1 + K_2) K_2 + \cdots) = \sum_{i=1}^{j} i p_i K_i T_i dx$$

where $T_i = K_1 + \cdots + K_i$. Hence

$$1 + \log(i) \propto i p_i K_i T_i$$

$$C(1 + \log(i))(i p_i)^{-1} = (T_i - T_{i-1}) T_i$$

where C is a constant. From the quadratic formula we get

$$T_i = \frac{T_{i-1} + \sqrt{T_{i-1}^2 + 4C(1 + \log(i))(i p_i)^{-1}}}{2} \approx T_{i-1} + \frac{2C(1 + \log(i))(i p_i)^{-1}}{T_{i-1}}$$

meaning

$$K_i = T_i - T_{i-j} \propto \frac{(1 + \log(i))(i p_i)^{-1}}{T_{i-1}}$$

5 Card game strategy

The AI used for the enemy is simple, choosing a card uniformly at random. Surprisingly, the optimal gameplay strategy can still be quite deep. Simulations quickly ruled out a myopic strategy which chose the card which maximized the probability of winning the next round.

We can investigate how the optimal strategy might look by case studies of individual hands. Given a player's hand and the enemy's deck, there is an optimal strategy for how to play the hand. The optimal strategy for the duel just consists of the collection of optimal strategies for each hand. Thus, we can study the impact of strategy by sampling hands and studying each hand in detail.

The idea to adopt a rock-paper-scissors type system was motivated based on findings from strategy simulations. Originally, the plan was to generate the matrix of card interactions completely by random. However, given the complexity constraints discussed above, this required most card pairs to be tied. But strategy simulations showed that this decreased the difference between optimal strategy and randomized strategy.

6 Challenges

Challenges consist of a series of enemies with randomized decks, all of whom must be defeated.

Since each challenge unlocks a card set, controlling access to challenges is vital. We have each challenge unlock two new challenges. Challenge 1 unlocks challenges 2 and 3, challenge 2 unlocks 4 and 5, challenge 3 unlocks 6 and 7, etc. The player is given a rating for each challenge: the formula is

$$rating = \max_{0 \le i \le j \le n} average(X_i, \ldots, X_j) + \log_2 \log_{10}(j - i + 1)$$

The challenge threshold for challenge i is set to

$$p_{crit,i} + \log_2 \log_{10} N_i$$

where N_i is the number of times a "typical" player would have to beat the challenge to unlock the new challenges.

Some asymptotic phenomena allow us to estimate a reasonable win rate for a challenge $p_{crit,i}$, even as the player's card collection grows, using simulations with fixed numbers of cards.

We set the card set to have p_i of off-type interactions. Supposing the challenge consists of $k_{i,1}$ "relevant" enemies using decks of size $k_{i,2}$, the distribution of off-type interactions (adding +1 when the player's card counters the enemy's card, and -1 vice versa) is $N(0, p_i/k_i)$ for p_i small, where $k_i = k_{i,1}k_{i,2}$. Next, we expect that the player's deck for the challenge will consist of cards with a high average probability to beat a random card from a random enemy in the challenge. Suppose the player has n cards. Then the number of cards the player has with an average $t = 0.1$ interaction is $n\bar{\Phi}(t\sqrt{k/p})$. Supposing the number of cards the player has at the ith challenge is Ci, we can scale p_i and k_i to keep $Ci\bar{\Phi}(t\sqrt{k/p})$ constant as i increases. The asymptotic phenomenon is that C becomes unimportant as i increases. In fact, due to the property of the normal tail, the number of the player's cards with average interaction more than $1.001t$ goes to zero. On the other hand, the number of the player's cards with average interaction greater than $0.999t$ is $O(\log i)$, and among these, the fractions of interactions which are positive goes to one.

Because of these properties, we can use finite-size simulations to predict what will happen asymptotically, at least for a fixed number of enemies. Since the player eventually gains access to every possible card with average interaction equal to t, we can construct the player's ideal deck by optimizing each card in the deck with the constraint of having average interaction equal to t, and then characterize the distribution of the win rate of the optimal deck for a random challenge. As the enemy's deck size increases, we expect that the details of how to optimize each card only end up depending on how much interaction t to allocate to each enemy. Given this information, we choose $p_{crit,i}$ so that the probability that the optimal win rate exceeds $p_{crit,i}$ is some probability p_{branch} greater than $1/2$. We can now view the game as a branching process: with probability p_{branch}, each challenge gives rise to two new challenges. Then the probability of ever getting stuck is a small constant depending on p_{branch}. Of course, due to our scoring system, a player can always get unstuck by brute forcing an astronomical number of attempts.

Since our approximation may not work well in the early stages, we choose N_i to be small for the early stages; we can cap N_i at say, 1000.