Relatório técnico da POC I

Simulações estatísticas no contexto de gachapons

Pesquisa Tecnológica

Orientador
Pedro Olmo Stancioli Vaz de Melo

Aluno
Lucas Rios Bicalho
2020006779

Departamento de Ciências da Computação
Universidade Federal de Minas Gerais

1. Introdução

No contexto de jogos online, um fenômeno recente e bastante criticado é a inserção de microtransações, lootboxes e afins, de modo a incentivar os jogadores a gastarem dinheiro real ao colocá-los em posição semelhante à de um apostador em um cassino online.

Entretanto, existe um gênero de jogos onde esse tipo de comportamento não só é normalizado como também é o atrator principal: são os gachapons, ou simplesmente gachas. A palavra tem sua origem no japonês 嘎查 (gāchá), se referindo originalmente às maquininhas de vendas de brinquedos nos quais são inseridas uma moeda e retira-se um prêmio aleatório.

Nesse tipo de jogo, o usuário depende fortemente de sorte para conseguir atingir algum objetivo específico: seja desbloquear um personagem que ele deseja, melhorias para os personagens que já tem ou avançar em estágios do jogo, em algum momento se deparará com uma barreira, onde precisará trocar uma certa quantidade de moedas do jogo por uma chance de conseguir o que deseja.

O modelo de negócios se torna bastante simples: uma vez que o usuário esgota as moedas que conseguiu jogando, existirá uma chance de que não tenha atingido seus objetivos. Nesse momento, precisará escolher entre desistir ou gastar dinheiro real para ter mais tentativas. Além disso, como as probabilidades tendem a ser relativamente complexas, com várias variáveis afetando o resultado da “loteria”, cria-se uma barreira, impedindo (ou desestimulando bastante) que o jogador tome uma decisão completamente informada.

Este gênero inicialmente se destacava em um nicho bem específico: o mercado de games mobile asiático. Entretanto, com o lançamento e sucesso generalizado de Genshin Impact, jogadores do mundo todo foram introduzidos a ele, não se restringindo mais apenas ao nicho original.

O objetivo do trabalho é desenvolver uma ferramenta que auxilie o jogador deste gacha em específico a visualizar melhor as chances que ele tem de obter os itens que deseja e julgar, racionalmente, se vale a pena ou não gastar dinheiro real.

A seguir, será apresentado a estrutura do código responsável pelo cálculo dessa futura ferramenta e as decisões de projeto por trás dela.

2. Referencial teórico

Genshin Impact é um jogo de gacha online, lançado em 2020 para PC, PlayStation 4, Android e iOS. É um dos maiores casos de sucesso deste gênero: de acordo com estimativas do site activeplayer.io, consistentemente consegue mais de 60 milhões de jogadores por mês.

Neste jogo, o sistema de gacha é utilizado para conseguir dois recursos: personagens e armas. Os personagens são classificados entre cinco e quatro estrelas, enquanto as armas são classificadas entre cinco, quatro e três estrelas. A raridade dos itens são proporcionais à sua classificação.

As principais moedas relevantes para o sistema de gacha são as que se seguem:

- **Destino entrelaçado**: moeda principal do gacha. Um destino permite obter um único item no gacha.
- **Primogemas**: moeda obtida por completar o conteúdo do jogo. 160 primogemas podem ser convertidas em um Destino entrelaçado. As formas de se conseguir primogemas consistentemente são missões diárias (60 por dia) e comprando um item específico da loja, com dinheiro real (90 por dia).
- **Cristal Gênesis**: moeda obtida com dinheiro real. Um cristal gênesis pode ser convertido em uma primogema.
- **Brilho Estelar Abandonado**: moeda obtida por realizar tentativas no gacha. Prêmios diferentes dão quantidades diferentes desta moeda. 5 brilhos estelares podem ser convertidos em um destino entrelaçado.
- **Bênção da Lua Nova:** item da loja que, após ser comprado, fornece 300 cristais gênesis imediatamente e 90 primogemas por dia pelos próximos 30 dias.

Imagem 1: menu da loja onde é feito a conversão de primogemas para destinos entrelaçados

Imagem 2: menu da loja onde é feita a compra da moeda paga do jogo

Um mesmo personagem pode ser obtido até 7 vezes: a primeira vez desbloqueia o personagem para uso. As seis vezes em seguida as capacidades do personagem. Ao adquirir um personagem de quatro estrelas “incompleto” (ou seja, que ainda não foi adquirido 7 vezes), ganha-se 2 brilhos estelares. Ao adquirir um personagem quatro estrelas “completo”, ganha-se 5 brilhos estelares. Ao adquirir um personagem cinco estrelas “incompleto”, ganha-se 10 brilhos estelares. Ao adquirir um personagem cinco estrelas “completo”, ganha-se 25 brilhos estelares.

Uma mesma arma pode ser obtida múltiplas vezes. Ao adquirir uma arma quatro estrelas, ganha-se 2 brilhos estelares. Ao adquirir uma arma cinco estrelas, ganha-se 5 brilhos estelares.

O sistema de gacha é, também, dividido em “banners”, onde se seleciona o tipo de objeto de interesse: personagens ou armas. Cada banner aumenta a probabilidade de se conseguir um personagem ou arma em específico, mas têm duração limitada e, para a grande maioria dos personagens e armas, esperar pelo retorno do seu respectivo banner é a única forma de obter o item.

Um **banner de personagem** tem um único personagem cinco estrelas e três personagens quatro estrelas como itens em destaque. É garantido que, em 90 tentativas neste banner, pelo menos um dos itens será um personagem cinco estrelas (não necessariamente o personagem em destaque no banner: 50% de chance de ser, 50% de chance de não ser). É garantido que, caso o último personagem cinco estrelas ganho não seja o personagem do banner, o próximo personagem cinco estrelas será. É garantido que, a cada 10 tentativas no banner, pelo menos um item será um item quatro estrelas ou superior (podendo ser um personagem ou uma arma). É garantido que, caso o último item quatro estrelas ganho não seja um dos personagens quatro estrelas em destaque no banner, o

próximo item quatro estrelas será. Todas estas garantias se mantém quando o tempo do banner se esgota e o banner de personagem é alterado por outro.

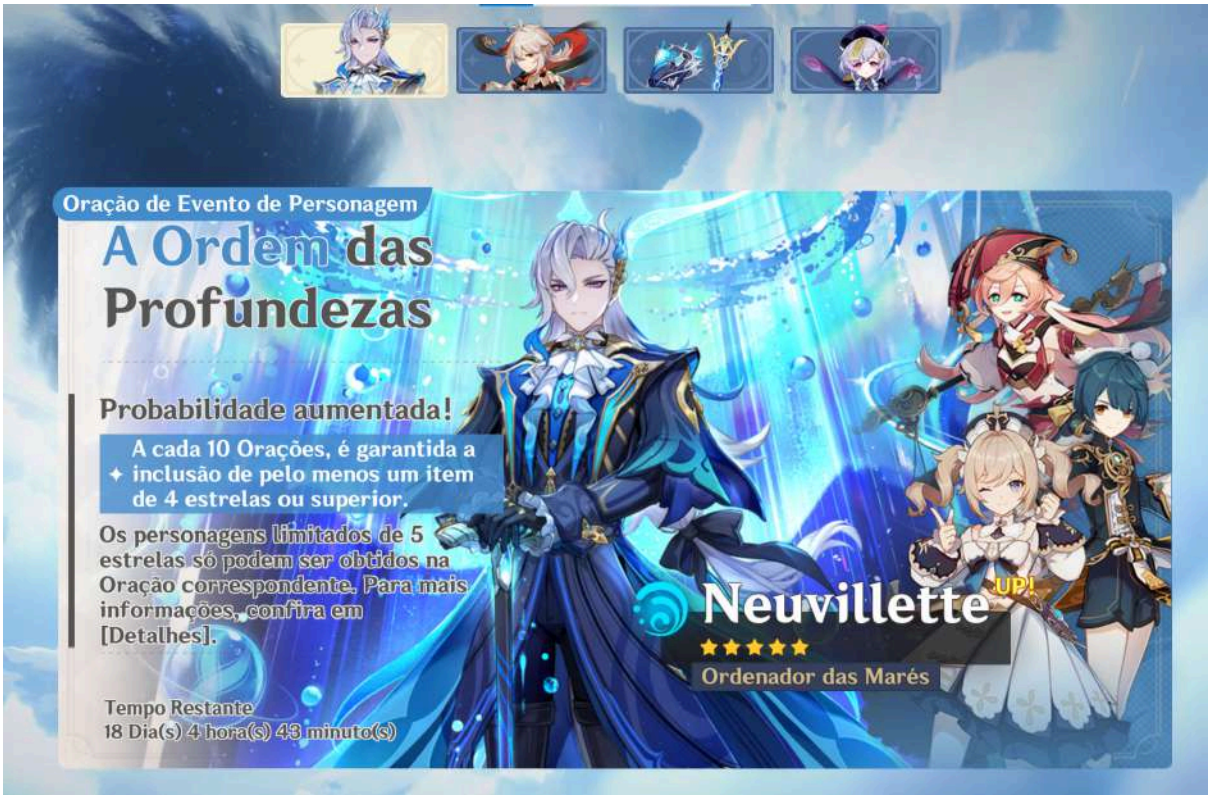

Imagem 3: exemplo de banner de personagem

Um **banner de arma** tem duas armas cinco estrelas e cinco armas quatro estrelas como itens em destaque. É garantido que, em 80 tentativas neste banner, pelo menos um dos itens será uma arma cinco estrelas (não necessariamente uma das armas em destaque no banner: 75% de chance de ser, 25% de chance de não ser). É garantido que, caso a última arma cinco estrelas ganha não seja uma das duas em destaque, a próxima arma será. É garantido que, a cada 10 tentativas no banner, pelo menos um item será um item quatro estrelas ou superior (podendo ser um personagem ou uma arma). É garantido que, caso o último item quatro estrelas ganho não seja uma das armas quatro estrelas em destaque no banner, o próximo item quatro estrelas será. Todas estas garantias se mantém quando o tempo do banner se esgota e o banner de arma é alterado por outro.

Como há duas armas cinco estrelas em destaque no banner, há a possibilidade de se conseguir múltiplas vezes uma arma cinco estrelas, mas nunca a que se deseja. Para remediar isso, foi implementado um sistema onde é possível selecionar uma arma em específico e, caso se ganhe duas armas cinco estrelas diferentes desta, a terceira será ela, garantidamente. Esta garantia **NÃO** se mantém quando o tempo do banner se esgota: uma vez que é trocado o banner, este contador é reiniciado, sendo necessário tirar mais duas outras armas para se obter a garantia novamente.

Imagem 4: exemplo de banner de arma

O banner de arma costuma ser sincronizado com os de personagens: ambos tendo duração de 21 dias, usualmente tem-se, nas armas cinco estrelas do banner, armas que sincronizam bem com os personagens cinco estrelas em destaque nos banners de personagem.

Oficialmente, as probabilidades disponibilizadas pela empresa de se obter um item em uma tentativa qualquer no gacha são as que se seguem:
Banner de personagens:

- 0.6% de probabilidade base para um personagem cinco estrelas, 1.6% de probabilidade conjunta (incluindo garantias)
- 5.1% de probabilidade base para um item quatro estrelas (2.55% personagens, 2.55% armas), 13% de probabilidade conjunta (incluindo garantias)

Banner de armas:

- 0.7% de probabilidade base para um personagem cinco estrelas, 1.85% de probabilidade conjunta (incluindo garantias)
- 6% de probabilidade base para um item quatro estrelas (3% personagens, 3% armas), 14.5% de probabilidade conjunta (incluindo garantias)

Entretanto, a comunidade do jogo descobriu, via mineração de dados em um banco de dados de 25 milhões de tentativas, que as probabilidades são um pouco melhores do que a empresa anuncia. Segundo o usuário Sengalev, a chance base de um personagem cinco estrelas no banner de personagem (0.6%) se mantém nas

tentativas 1 a 73, aumentando linearmente em 6% a cada tentativa acima disso (6.6% na 74, 12.6% na 75…) até chegar em 100% na tentativa 90.

Similarmente, a chance base de uma arma cinco estrelas no banner de arma (0.7%) se mantém nas tentativas 1 a 62, aumentando linearmente em 7% a cada tentativa acima disso (7.7% na 63, 14.7% na 64…) até chegar em 100% na tentativa 77.

Esses números foram publicados em Junho/2021 e, desde então, nenhuma alteração foi feita no sistema de probabilidades do jogo.

3. Implementação

O código pode ser dividido em três partes: **planejamento**, **funções auxiliares** e o **cálculo propriamente dito**.

O **planejamento** se refere à situação inicial do usuário, suas decisões e estratégias para conseguir os itens e o ganho de recursos/moedas ao longo deste processo. Esta parte está implementada no arquivo *gachaPlanning.py*.

Este arquivo serve para definir principalmente as cinco classes que se seguem:

- *gachaWishPlan*: classe que armazena o conjunto de todos os passos da estratégia de um usuário, e que será utilizada para calcular a sua probabilidade de sucesso.
- *bannerWishPlan*: classe abstrata que armazena um único passo da estratégia do usuário. Será especializadas pelas suas subclasses a seguir:
- *updateWishingSettings*: indica um passo da estratégia onde alguma informação é alterada: quantidade de gemas, de desejos, de brilho estelar, alteração no caminho epitomizado, etc. É qualquer decisão do usuário que não seja de fato gastar moedas no gacha.
- *updateWishingSettings*: indica um passo da estratégia onde o usuário realiza tentativas no banner de personagem limitado. Recebe os nomes dos quatro personagens como identificador (na falta deles, insere um nome genérico) e permite adotar tanto estratégias de “realizar tentativas o número de tentativas superar N” quanto “realizar tentativas até conseguir N cópias do personagem”.
- *weaponBannerWishPlan*: indica um passo da estratégia onde o usuário realiza tentativas no banner de armas. Recebe o nome das sete armas como identificador (na falta deles, insere um nome genérico) e permite adotar tanto estratégias de “realizar tentativas o número de tentativas superar N” quanto “realizar tentativas até conseguir N cópias de uma arma”.

As **funções auxiliares** se referem ao final da seção anterior deste documento: quais são as chances de sucesso em uma tentativa em específico. Em outras palavras, é a parte do código que gera a informação “dado que já fizemos n tentativas no banner de personagem limitado, qual a chance de um sucesso na tentativa n+1?

Contém quatro funções para calcular probabilidades:

- *getLimitedBannerOddsFiveStar*
- *getLimitedBannerOddsFourStar*
- *getWeaponBannerOddsFiveStar*
- *getWeaponBannerOddsFourStar*

Além disso, contém duas funções para simulação de uma tentativa em um banner:

- *doPullLimited*
- *doPullWeapon*

Estas funções usam um random number generator para, dada a quantidade de roletas já realizadas em um banner (de personagem limitado ou de armas), decidir se esta roleta foi de um item 5 estrelas, 4 estrelas ou 3 estrelas.

Por fim, o **cálculo propriamente dito** foi implementado por dois caminhos: buscando calcular todos os estados possíveis para uma dada estratégia, ou realizando múltiplas simulações da estratégia e analisando a amostra disponível.

O primeiro caminho não deu frutos: mesmo em uma versão relaxada do problema (onde só se consideravam itens 5, 4 ou 3 estrelas, sem distinção entre itens de mesma raridade), ocorria uma explosão exponencial do número de estados possíveis. Tentar contornar essa explosão via podar os estados com probabilidade arbitrariamente pequena de ocorrer leva a dois casos indesejáveis:

- Se a poda ocorrer em estados com uma chance muito baixa, ela não poda o suficiente, e a explosão exponencial ainda ocorre. Programa não retorna nada, pois fica eternamente processando estes estados.
- Se a poda ocorrer em estados com uma chance mais alta, o programa eventualmente termina, mas a soma dos estados podados se torna muito alta, fazendo com que o resultado tenha muita pouca informação útil a ser exibida para o usuário.

Este caminho só é viável para quantidades pequenas (nos testes empíricos, até treze) de tentativas totais, o que é muito inferior à quantidade de roletas que o usuário médio do sistema usará em sua estratégia.

O código referente a esse caminho está no arquivo *gachaCalculation.py*, mas ele não é utilizado no resto do sistema.

O segundo caminho envolve fazer n simulações da estratégia delimitada pelo usuário e retornar os resultados dela. Se n for grande o suficiente, as médias das simulações deverão convergir para a probabilidade real do usuário.

Para isto, duas classes foram implementadas:

- *gachaSimulation*: representa uma única execução da estratégia delimitada pelo usuário, do início ao fim. Recebe como parâmetros de inicialização a classe *gachaWishPlan* contendo a estratégia, uma função de testes delimitando quais são as variáveis (personagens e armas) de interesse do usuário e a situação inicial do usuário.

  Contém variáveis internas responsáveis por armazenar a quantidade de cada

personagem, arma e recursos relevantes para a simulação.
Uma de suas duas principais funções, *doSimulation* verificar e segue cada um dos passos delimitados dentro da classe *gachaWishPlan*, o que alterará as variáveis internas mencionadas anteriormente.
A sua outra função principal, *doTestFunction*, aplica a função de teste recebida na inicialização sobre as variáveis internas da simulação, permitindo que sejam feitas perguntas do tipo “Ao final da simulação, a quantidade de vezes que consegui um personagem X foi superior a 2?”, retornando True ou False para isto.

- *multipleGachaSimulations*: representa a execução múltipla da classe anterior. Também recebe como parâmetros de inicialização a estratégia, a situação inicial do usuário e a função de teste. Sua principal função, *doSimulations*, serve para criar n instâncias da classe anterior, realizar sua simulação, executar a função de testes e armazenar o resultado. Após isto, retorna o resultado agregado da função de testes para todas as n simulações. Ou seja, ao invés de responder True ou False para a indagação “Ao final da simulação, a quantidade de vezes que consegui um personagem X foi superior a 2?”, esta classe retorna a quantidade (ou a fração da quantidade) de simulações para as quais esta indagação retornou True.

  Este segundo caminho está implementado no arquivo *gachaSimulation.py*.

Um exemplo de uso do sistema está no arquivo *endUser.py*: cria-se uma instância de *gachaWishPlan* e adiciona-se todos os passos planejados da simulação, como o ganho de moedas, o gasto delas em personagens e o gasto delas em armas. Essas adições são feitas com as subclasses de *bannerWishPlan* mencionadas anteriormente.

Cria-se também a função de testes, com a possibilidade de nomear cada teste e também de usar operações booleanas em um mesmo teste (foi útil em um dos exemplos que será exposto mais adiante).

Insere-se, opcionalmente, a lista de todos os personagens que o usuário tem e, a partir disto, pode-se obter o resultado da testagem da função anterior a partir de um número arbitrário de simulações. Este resultado é então impresso, com um intervalo de confiança de 95%.

4. Resultados e análise

O resultado da POC I foi bem-sucedido: implementar um algoritmo útil e eficiente para calcular as probabilidades, para um jogador de Genshin Impact, de conseguir desbloquear os itens de seu interesse.

Em *endUser.py*, está a simulação para a minha própria conta, com valores reais, para o cálculo da probabilidade de desbloquear a personagem Furina (C0) após a compra de um pacote de 3280 gemas, com o bônus de primeira compra. Nas funções de teste, testou-se cada uma das 6 cópias possíveis (C1-C6). O resultado, realizando 10.000 iterações, foi obtido em menos de 2 segundos, com intervalo de confiança de menos de um ponto percentual.

```
Furina at least at C0: 100.00% ± 0.00%
Furina at least at C1: 100.00% ± 0.00%
Furina at least at C2: 78.17% ± 0.81%
Furina at least at C3: 22.71% ± 0.82%
Furina at least at C4: 1.85% ± 0.26%
Furina at least at C5: 0.14% ± 0.07%
Furina at least at C6: 0.00% ± 0.00%
At least one copy of furina's weapon: 0.00% ± 0.00%

real    0m1.772s
user    0m1.752s
sys     0m0.001s
```

Imagem 5: resultado da simulação

Este intervalo já é suficiente para um jogador determinar se a estratégia vale a pena ou não: se ela, por exemplo, retornar 20% sob este intervalo, não faz realmente diferença prática na decisão se a chance real é 19%, 21% ou qualquer valor no meio.

Além disso, também foi realizado um teste de stress, em *endUserStress.py*: para o caso onde o número de tentativas no gacha se aproxima de 1.000, realizar 10.000 simulações aumenta o tempo de execução para 15 segundos. Casos acima deste número não precisam ser considerados, pois são quantias que muito raramente acontecem (o jogador médio não chega a ter nem perto de 1.000 desejos, só acontecendo em casos que gastam quantidades grandes de dinheiro).

Por fim, um teste prático em pequena escala foi feito, em *endUserE.py*, *endUserT.py* e *endUserG.py*. Foi possível testar, para três jogadores diferentes, as probabilidades de seu interesse:

- Usuário G desejava saber quais chances tinha de conseguir a personagem Arlecchino, sua arma e seis cópias do personagem, se tentasse obtê-los exatamente nessa ordem. A simulação revelou que tinha aproximadamente 50% de chance de pegar a personagem, mas quase zero de conseguir a arma ou cópias da personagem.
- Usuário T buscava o personagem Lyney, duas cópias e sua arma, mas não necessariamente nesta ordem. Fizemos quatro simulações (buscando a arma após pegar o personagem, após a primeira cópia, após a segunda cópia e desistindo da arma), e o usuário, ao olhar todas as probabilidades, acabou decidindo priorizar suas chances com as cópias e desistir completamente da arma.
- Usuário E desejou, após ter gasto 100 tentativas em um banner tentando conseguir a sexta cópia da personagem Faruzan (e só ter conseguido até a quarta), saber o quão azarado ele tinha sido. Relatou que, inicialmente, achava que tinha sido MUITO azarado, mas pelo resultado da simulação, a probabilidade de ter acontecido isto era de aproximadamente 25%, revelando que tinha uma expectativa errônea sobre suas chances.

Este teste prático revelou que o programa de fato está sendo capaz de trazer resultados úteis que não estão prontamente disponíveis no jogo ou em ferramentas populares já existentes.

5. Próximos passos

Para o próximo semestre, o objetivo é utilizar esta base de código para implementar um website e tornar público o acesso à ferramenta. Isto será facilitado pela existência de bibliotecas, como o Streamlit, que permitem rápida prototipação e deployment. Adicionalmente, a atual estruturação da etapa de planejamento da estratégia em múltiplas classes que encapsulam a interação com a classe de simulação tornará mais simples este processo,

6. Referências bibliográficas

GENSHIN Impact. Versão 4.5. Jurong East, Singapore: HoYoVerse, 2020. Jogo eletrônico. Disponível em: www.genshin.hoyoverse.com. Acesso em: 1 abr. 2024.

HOYOLAB: Statistical model for Genshin Impact's droprates. [*S. l.*], 28 jun. 2021. Disponível em: www.hoyolab.com/article/497840. Acesso em: 1 abr. 2024.

GENSHIN Impact Live Player Count and Statistics. [*S. l.*]. Disponível em: www.activeplayer.io/genshin-impact. Acesso em: 1 abr. 2024.

GACHA: definition in the Cambridge English Dictionary. [S. l.], 5 abr. 2024. Disponível em: www.dictionary.cambridge.org/us/dictionary/english/gacha. Acesso em: 1 abr. 2024.