Flairplay Engine: Uma Ferramenta para Apoio ao Game Designer em Balanceamento de Jogos Digitais

Flairplay Engine: A Tool to Support Game Designers in Digital Game Balancing

Maria Clarissa Carminatti[1], Marcus Parreiras[1,2], Leonardo Ruma Martins[2], Lucas Vieira Dinato[2], Marcelo Silva[1], Geraldo Bonorino Xexéo[2]

[1] COENP - Coordenadoria de Engenharia de Produção
Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ)

[2]LUDES - Programa de Engenharia de Sistemas e Computação
COPPE - Universidade Federal do Rio de Janeiro
Avenida Horácio Macedo, 2030, CT, Bloco H, sala 319, Rio de Janeiro, RJ - Brasil

{maria.assumpcao}@aluno.cefet-rj.br, {lucasvdinato, leonardo.ruma}@gmail.com,
{marcelo.rodrigues}@cefet-rj.br, {mparreiras, xexeo}@cos.ufrj.br

Abstract. Introduction*: Game balancing is a continuous challenge in the industry, often lacking quantitative evaluation methods to support the design process.* ***Objective:*** *The objective of this paper is to present the Flairplay Engine, a tool developed to assist game designers in balancing digital games through objective metrics derived from telemetry data.* ***Methodology or Steps****: We implemented a telemetry model in a 2D Roguelike game built in Unity, processing the collected JSON data through a full-stack web architecture (Vue.js and Spring Boot), and calculating specific metrics based on the DRAGO thesis (Balance, Permanence, Uncertainty, Decisive Advantage, Drama, and Entropy).* ***Expected Results:*** *The platform successfully processed player data, generating visual and quantitative charts that allowed the clear identification of dominant strategies and the precise impact of random elements on gameplay, proving its efficacy as an analytical support tool for decision-making.*
Keywords *Balancing, Game Design, Digital Games, Telemetry, Metrics.*

Resumo. Introdução: *O balanceamento de jogos é um desafio contínuo na indústria, frequentemente carecendo de métodos quantitativos de avaliação que apoiem o processo de design.* ***Objetivo:*** *Este artigo visa apresentar a Flairplay Engine, uma ferramenta focada em apoiar game designers na análise do balanceamento de jogos por meio do processamento de dados reais de telemetria.* ***Metodologia ou Etapas****: Foram aplicados algoritmos de extração de dados em um jogo 2D Roguelike desenvolvido em Unity. Os dados coletados foram enviados para uma arquitetura web onde métricas matemáticas baseadas na tese DRAGO (Equilíbrio, Permanência, Incerteza, Vantagem Decisiva, Drama e Entropia) foram processadas e transformadas em visualizações analíticas.* ***Resultados Esperados:*** *A ferramenta permitiu a identificação clara de estratégias dominantes, avaliação de incerteza e análise de drama, fornecendo ao designer uma visão objetiva do comportamento dos jogadores, comprovando sua eficácia como ferramenta de suporte analítico.*
Palavras-Chave *Balanceamento, Game Design, Jogos Digitais, Telemetria, Métricas.*

1. Introdução

A indústria de jogos eletrônicos consolidou-se, nas últimas décadas, como o maior segmento da indústria do entretenimento global, superando historicamente as receitas somadas do cinema e da música [Geloneze e Arielo 2017]. Para sustentar esse crescimento exponencial e manter a retenção de público, os desenvolvedores enfrentam o desafio contínuo de criar produtos que justifiquem o investimento de tempo e dinheiro por parte dos jogadores. Inúmeros fatores ditam o sucesso de um título, incluindo a fluidez técnica, a direção de arte e as campanhas de marketing. Contudo, o balanceamento de jogo destaca-se como um dos pilares mais críticos da experiência do usuário (UX), impactando diretamente o engajamento e a viabilidade a longo prazo de um projeto.

O balanceamento consiste em garantir que o ecossistema do jogo ofereça uma experiência justa, onde o nível de desafio seja instigante o suficiente para evitar o tédio, mas não tão punitivo a ponto de gerar frustração aguda [Filho et al. 2021]. Em contextos competitivos, o balanceamento dita a viabilidade de táticas e personagens; em jogos educacionais, assegura que as barreiras cognitivas estejam alinhadas ao processo de aprendizado; em experiências casuais, atua na maximização da diversão. Um sistema desbalanceado não apenas consome recursos de desenvolvimento desnecessariamente, mas pode colapsar todo o conjunto de regras concebidas, culminando no abandono prematuro do produto pelos usuários.

Apesar de sua importância inegável, a disciplina de balanceamento ainda carece de processos padronizados e frequentemente depende de intuição, heurísticas empíricas ou longas e custosas sessões de testes manuais (Quality Assurance). A ausência de ferramentas que traduzam o comportamento dos jogadores em dados acionáveis e métricas objetivas torna o ciclo de design demorado e suscetível a vieses humanos [Becker e Görlich 2020]. Desenvolvedores independentes e estúdios de médio porte, em particular, sofrem com a falta de infraestrutura de *analytics* para identificar gargalos de design estrutural.

Diante desta lacuna tecnológica e metodológica, este trabalho apresenta a Flairplay Engine, uma ferramenta web desenvolvida com o intuito de democratizar e quantificar o processo de balanceamento de jogos digitais. O artefato proposto coleta, armazena e processa dados de telemetria reais registros automatizados de eventos do jogo e aplica um conjunto de algoritmos matemáticos baseados na tese DRAGO [Parreiras 2025]. A plataforma não substitui a figura do *game designer*, mas atua como um sistema de suporte à decisão, devolvendo métricas estruturadas de Equilíbrio, Permanência, Incerteza, Vantagem Decisiva, Drama e Entropia.

Para validar a arquitetura e a precisão algorítmica da ferramenta, conduzimos um estudo de caso prático utilizando um jogo do gênero Roguelike 2D desenvolvido na *engine* Unity. Através da injeção de rotinas de telemetria personalizadas, capturamos metadados complexos sobre a movimentação, tomada de dano, coleta de itens e exploração espacial por parte dos jogadores. Os resultados demonstram a robustez da plataforma ao evidenciar discrepâncias e dominâncias de estratégias de maneira estritamente factual.

A estrutura deste artigo foi pensada para guiar o leitor desde os conceitos basilares até a implantação sistêmica: a Seção 2 consolida a revisão da literatura sobre telemetria e as métricas utilizadas; a Seção 3 detalha a metodologia de construção da ferramenta e

os algoritmos aplicados; as Seções seguintes irão discorrer sobre os dados coletados e as perspectivas de evolução da plataforma.

2. Revisão da Literatura

Para sustentar a arquitetura e os algoritmos da Flairplay Engine, fundamentamos nosso trabalho em vertentes consagradas e contemporâneas do *Game Design*, telemetria de sistemas interativos e modelagem matemática de comportamento.

2.1. O Ecossistema de Jogos Digitais e o Gênero Roguelike

O jogo digital transcende o entretenimento, caracterizando-se como um sistema complexo governado por regras, onde jogadores engajam em conflitos artificiais gerando resultados quantificáveis [Salen e Zimmerman 2003]. A interatividade digital permite a alteração assíncrona ou síncrona do estado do sistema por meio de *inputs* do usuário. No contexto do *Game Design*, o desenvolvimento aglutina múltiplas competências (arte, som, programação, *quality assurance*), e a taxonomia dos jogos costuma ser atrelada aos tipos de desafios propostos [Adams 2013].

Nosso estudo de caso foca na categoria de jogos *Roguelike*. Originado na década de 1980 com o jogo *Rogue*, este gênero é marcado por características rigorosas: Geração Processual de Níveis (garantindo alta entropia inicial e rejogabilidade), Morte Permanente (*permadeath*, elevando as métricas de Drama e Vantagem Decisiva), e altíssima complexidade sistêmica e tática [Harris 2009, Craddock 2015]. Em virtude da impossibilidade de decorar padrões estáticos (como ocorre em jogos clássicos de plataforma), o balanceamento em *Roguelikes* recai intensamente sobre o ajuste das matrizes de probabilidade e nos custos de oportunidade que o jogador enfrenta.

2.2. Telemetria Aplicada

A telemetria de jogos corresponde ao registro automático, remoto e estruturado de pontos de dados relativos a ações e eventos sistêmicos. Ferramentas modernas de telemetria superam metodologias tradicionais de pesquisa (como questionários e observação não intrusiva), oferecendo amostras de dados puramente factuais, mitigações contra a falha de memória do jogador e insights diretos sobre a ergonomia e o fluxo do jogo [Rashed et al. 2025]. A injeção de *trackers* baseados em eventos como coleta de um item de regeneração de vida ou a movimentação espacial no grid permite o mapeamento preciso de progressão geométrica de fases, como abordado e modelado na infraestrutura adotada em nosso backend [David e Castro 2019].

2.3. Métricas de Balanceamento DRAGO

O conceito de balanceamento, embora universal, não possui uma "receita"única, dependendo fortemente do gênero [Becker e Görlich 2020]. A base teórica das funções desenvolvidas na Flairplay Engine deriva diretamente da tese DRAGO (Design de Regras Avaliativas para Gameplays Otimizados) de Parreiras [Parreiras 2025], que propõe uma taxonomia padronizada. Para a formalização, assumimos os estados de "Vitória"(avanço favorável de estado), "Derrota"e "Liderança"(vantagem provisória). As definições conceituais estão descritas na Tabela 1.

A taxonomia DRAGO introduz seis vetores primários que nossa arquitetura implementa:

Tabela 1. Definições de conceitos sistêmicos em um jogo.

Conceito	Definição Sistêmica	Exemplo Prático
Fase	Conclusão de um nível ou etapa discreta e enumerável do jogo.	Um "dia"sobrevivido no Roguelike.
Partida	Superconjunto de fases que inicia no começo do sistema e encerra com vitória ou derrota final.	A execução contínua (run) até o Game Over.
Estratégia	Grafo de decisões heurísticas visando maximizar o vetor de vitória.	"Focar apenas em coletar itens de cura".
Liderança	O estado momentâneo de vantagem direcional em relação à falha sistêmica.	Terminar o turno com mais HP do que iniciou.

Fonte: Adaptado de [Parreiras 2025]

- **Equilíbrio**: A razão normalizada da viabilidade de sobrevivência de diferentes estratégias submetidas ao mesmo ambiente estocástico.
- **Permanência**: A quantificação temporal (em números discretos de fases/turnos) que um jogador consegue sustentar o estado de liderança antes de uma quebra de ritmo.
- **Incerteza**: A taxa de volatilidade da Liderança (deriva da contagem de *swaps* de vantagem pelo total de iterações do sistema).
- **Vantagem e Momento Decisivos**: O limiar numérico de estado (HP_{lim}) onde a curva de derrota cai para zero por aproximação, e o *timestamp* (momento) em que isso ocorre.
- **Drama**: Aplicação do Princípio de Pareto para mapear as reviravoltas; mede o cluster de 20% de piores estados iniciais que ainda assim culminaram em vitória.
- **Entropia**: Modelagem preditiva baseada em Correlação de Pearson para medir o peso de variáveis estocásticas (como distância de geração de inimigos) versus o sucesso probabilístico.

2.4. Trabalhos Correlatos

A literatura aponta diferentes rotas para automatizar o balanceamento. A tese *Pegasus* [Silva 2018] utiliza simuladores paramétricos restritos ao *designer* para prever a quebra de desafios progressivos de acordo com *inputs* hipotéticos. Trabalhos como os de Jaffe et al. [Jaffe et al. 2012] ("Evaluating Competitive Game Balance with Restricted Play") utilizam agentes de Inteligência Artificial para operar matrizes de força bruta e comportamento "guloso"(*greedy*) contra comportamentos exploratórios em testes de Monte Carlo, apontando desvios numéricos. Semelhantemente, ferramentas preditivas aplicadas a RPGs de mesa, como simuladores de *Dungeons & Dragons* [Fonseca 2024], cruzam planilhas de atributos estatísticos para informar antecipadamente o Índice de Desafio (CR) para Mestres de Jogo.

A Flairplay Engine diverge desses trabalhos ao não depender de agentes de IA simulados ou simulação paramétrica estática, mas de sessões reais providas de *inputs* orgânicos, lidando com a latência humana, falhas mecânicas e os erros de heurística não previstos por computadores. Nossa ferramenta não prevê o balanceamento; ela prova o (des)balanceamento através do retrocesso do log histórico em larga escala.

3. Metodologia

O ciclo metodológico englobou a instanciação do *tracker* de telemetria dentro da *engine* do jogo-alvo, a construção da plataforma analítica (arquitetura cliente-servidor) e a transcrição teórica das métricas para algoritmos de processamento de dados (ETL).

3.1. Instrumentação da Telemetria

Utilizamos como baseline o modelo de telemetria open-source desenvolvido por David e Castro [David e Castro 2019]. A *engine* utilizada foi a Unity, parametrizada em C#. O rastreamento foi acoplado através da técnica de interceptação de eventos (Event Dispatching). Os dados encapsulados compreendiam coordenadas geométricas (x, y), carimbos de tempo, delta de vida (*playerLife*) e tipagem do evento (*Spawn*, *Damage*, *Heal*, *GameOver*).

Estes artefatos eram submetidos de forma assíncrona (visando não penalizar a taxa de quadros por segundo do jogo) para uma infraestrutura noestuturada em tempo real (Firebase Realtime Database), cujo dump era extraído em uma matriz no padrão *JSON (JavaScript Object Notation)*.

```
"users": {
    "-OLBmJkJ8jqadKgp4-B2": {
        "info": {
            "sessionDuration": 1032.53369
        },
        "rounds": [
            {
                "duration": 8.574003,
                "nodes": [
                    {
                        "id": 0,
                        "info": {
                            "playerLife": 100
                        },
                        "link": -1,
                        "name": "start",
                        "position": {
                            "x": 0,
                            "y": 0,
                            "z": 0
                        },
                        "time": 2.869811,
                        "type": "Single Event"
                    },
...
```

Figura 1. Código – (JSON) Estrutura de Dados da Telemetria.

3.2. Arquitetura do Sistema e Normalização

A plataforma web foi desenhada sob um paradigma de microsserviços monolitizados, orquestrada pelo gerador JHipster.

- **Front-end**: Construído em Vue.js suportado por TypeScript, garantindo validações de tipagem fortes no envio de dados de formulário e filtros complexos de paginação. Gráficos dinâmicos foram renderizados em canvas utilizando a biblioteca *Chart.js*.
- **Back-end**: Motor processual em Java 17 acoplado ao ecossistema *Spring Boot*. Responsável por fazer o *parsing* do JSON de *n* Megabytes via *Jackson*, mapeando os nós para classes de Entidade Orientadas a Objeto.
- **Banco de Dados e Deploy**: Camada de persistência relacional gerida pelo PostgreSQL em produção (com H2 Database em dev), empacotado via Docker e hospedado nos contêineres do provedor em nuvem Render.

O fluxo de dados normaliza o arquivo bruto importado e destrincha o nó ‘users‘ em registros discretos na tabela ‘Partida‘, atrelando uma relação de multiplicidade com a tabela genérica ‘Estrategia‘. O Modelo de Entidade Relacionamento (MER) suporta chaves estrangeiras que acoplam Arquivos, Usuários e Autenticação.

3.3. Implementação Algorítmica das Métricas

O cálculo demandou que as equações analíticas fossem traduzidas para lógicas operacionais nas instâncias ‘MetricaService‘ no *Spring Boot*.

1. Equilíbrio: O algoritmo processa a agregação do *Win Rate* por tipo de Estratégia (e). Sendo p_e o percentual iterativo de vitórias de uma estratégia e μ_p a média global, o grau de desequilíbrio isolado D_e é expresso pela razão ajustada:

$$D_e = \frac{p_e - \mu_p}{\mu_p} = Np_e - 1 \tag{1}$$

2. Permanência e Mitigação de Outliers: Através de varreduras em janelas deslizantes nas listas de fases, calculamos o máximo intervalo contínuo de *flags* verdadeiras de vitória. Para evitar a poluição de métricas através de dados de "jogadores excepcionais"ou "afk"(*Away from Keyboard*), implementou-se o filtro do Intervalo Interquartil ($IQR = Q3 - Q1$), rejeitando estritamente tuplas onde $Permanencia < (Q1 - 1.5 \cdot IQR)$ ou $Permanencia > (Q3 + 1.5 \cdot IQR)$.

3. Drama (Curva de Pareto): O serviço isola as tuplas de fases contendo a quantidade de *playerLife* abaixo da Vantagem Decisiva. Ele cria uma distribuição de frequência f_i, realiza a ordenação decrescente e aplica um somatório acumulativo D_{acum_i}. A renderização front-end então plota a linha limite de 80% de Pareto visualmente.

4. Entropia através do Algoritmo A^* e Heurística de Manhattan: Devido ao *grid* estrito da movimentação do roguelike (movimentos ortogonais, impedindo transições em hipotenusas), aplicamos a Geometria do Táxi (Distância de Manhattan).

$$h(x, y) = |x_{objetivo} - x| + |y_{objetivo} - y| \tag{2}$$

Para calcular o "Caminho Mínimo"livre de colisões com os inimigos processualmente gerados, imbuímos o backend de um *solver* A^* (A-Star) nativo. O *pathfinding* atribuiu custos marginais $g(n)$ reduzidos para a proximidade a itens de cura ("Foods"$\Delta = -10$, "Sodas"$\Delta = -20$) e custo impeditivo a nós contendo predadores ou paredes intransponíveis. O valor retornado da execução do grafo (‘roundScore‘) alimentou a Coeficiente de Correlação de Pearson ($\rho_{X,Y}$) em relação à classificação booleana do final da fase.

3.4. Declaração do uso de IA

Durante a elaboração deste trabalho, a inteligência artificial Gemini (Google) serviu como recurso auxiliar na fase de redação e formatação deste estudo. Sua atuação limitou-se à condensação textual para adequação de normas, estruturação tabular e transposição de referências para linguagem BibTeX. Os autores ratificam que a supervisão intelectual e a análise dos dados foram conduzidas sem interferência automatizada, sendo estes os únicos responsáveis pelo conteúdo e veracidade do manuscrito.

4. Resultados

A validação da Flairplay Engine ocorreu mediante a coleta de dados de 21 partidas completas do jogo 2D Roguelike, totalizando 405 fases jogadas por diferentes voluntários. Para a extração paramétrica do Equilíbrio, foram estipuladas quatro heurísticas de comportamento (estratégias) simuladas pelos jogadores: *Sempre em Frente* (movimentação estrita para cima e direita), *Coletar Itens* (priorização absoluta de *pickups* de cura), *Estratégia Padrão* (comportamento de cautela mista), e *Contar Passos* (cálculo antecipado da matriz de movimentação inimiga para evitar dano).

A análise global da Taxa de Vitórias revelou uma média de 19,29 fases (figura 2) por partida (Desvio Padrão $\sigma = 11,96$). A distribuição entre vitórias e derrotas manteve-se próxima, com uma ligeira vantagem para as derrotas (cerca de 10% de diferença), denotando uma dificuldade moderadamente elevada, o que é coerente com o design punitivo esperado de um *Roguelike*. A Tabela 2 sumariza o desempenho isolado de cada estratégia submetida à plataforma.

Tabela 2. Comparativo de Desempenho por Estratégia Adotada

Estratégia	Taxa de Vitórias (%)	Média de Fases	Desvio Padrão (σ)
Sempre em Frente	48,21%	11,20	5,91
Coletar Itens	48,68%	15,20	2,79
Estratégia Padrão	41,09%	25,00	4,07
Contar Passos	45,82%	64,60	8,33

Fonte: Elaborada pelo autor (2026)

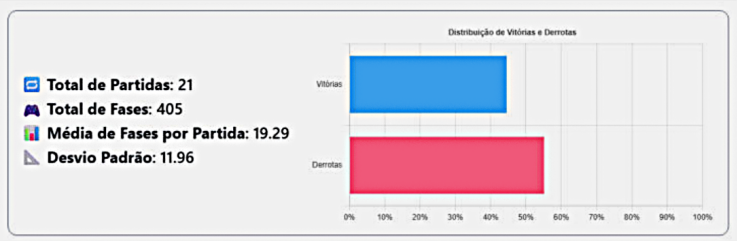

Figura 2. Resultados da métrica de Taxa de Vitórias.

No processamento da métrica de Equilíbrio, a plataforma evidenciou um desequilíbrio drástico e intencional (figura 3). A estratégia *Contar Passos* despontou como a Dominante, registrando um índice de desequilíbrio positivo superior a $+120\%$, enquanto todas as demais figuraram no espectro negativo. Isso ocorreu porque, ao prever a rota dos inimigos, o jogador otimizava a variável *playerLife*, alcançando a média expressiva de 64,60 fases sobrevividas por sessão.

No que tange à Permanência e Incerteza, os dados confirmaram a volatilidade do sistema. A Permanência Máxima global (sequência ininterrupta de vitórias) limitou-se a 6 fases (ou 8 fases ao incluir *outliers*), com uma mediana de 3. O índice de Incerteza fixou-se em 40,49%, revelando constantes trocas de liderança a cada nova transição de tela.

A Vantagem Decisiva foi calculada observando o "teto"de derrotas. O sistema apontou que jogadores que iniciavam uma fase com 106 pontos de vida ou mais não sofriam *Game*

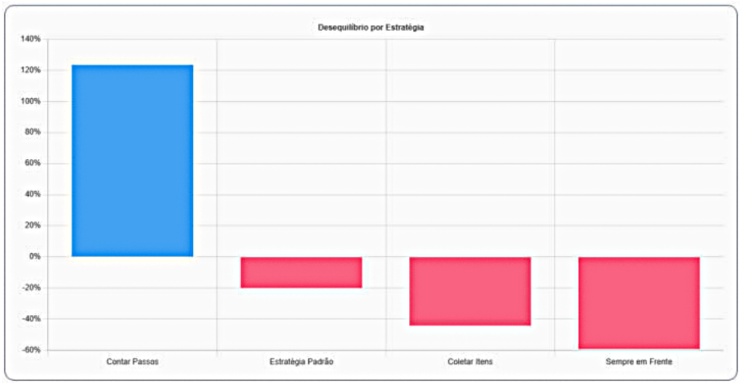

Figura 3. Gráfico de Desequilíbrio por Estratégia.

Over (figura 4). O Momento Decisivo foi identificado precocemente (Média = 1,42 fases), indicando que a segurança do jogador é consolidada logo no início da partida, a menos que ele sofra penalidades sucessivas por má gestão de risco. A métrica de **Drama**, baseada no Princípio de Pareto (figura 5), traçou o limite de 80% de risco acumulado na marca de 65 pontos de vida, identificando estas ocorrências como "Sucessos Improváveis"em caso de avanço de fase.

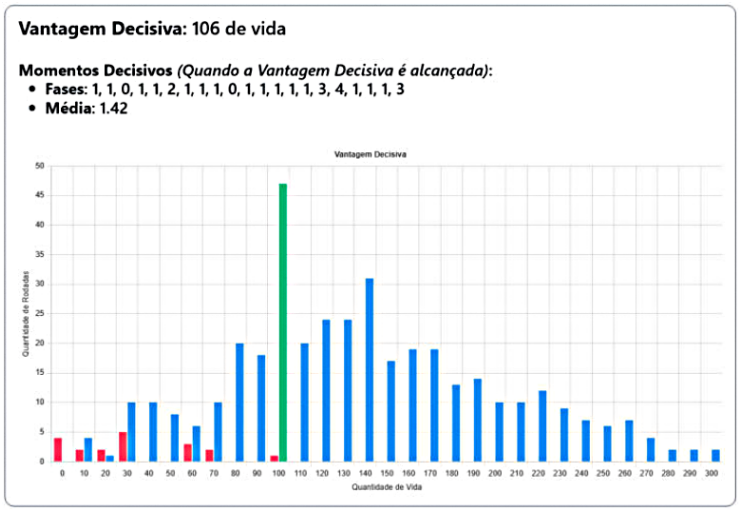

Figura 4. Resultados da métrica de Vantagem Decisiva.

Por fim, a modelagem de Entropia via Correlação de Pearson evidenciou quais componentes estocásticos ditaram o sucesso das *runs*. O "Caminho Mínimo"(calculado em *backend* pelo algoritmo A^* usando Distância de Manhattan [Ribeiro 2018]) e a quantidade de "Refrigerantes"(itens de cura maior) exibiram correlações positivas fortes (+35,1% e +37,5%, respectivamente). Em contrapartida, a densidade e proximidade geométrica de inimigos gerou entropia negativa de até -18,8%. A Entropia Geral média do sistema estacionou em 18,25%, validando que, embora a aleatoriedade afete o micro-jogo, ela não invalida a macro-estratégia.

5. Discussão

A experimentação empírica com a Flairplay Engine permite traçar correlações diretas entre a arquitetura de *software* desenvolvida e a teoria analítica de *Game Design*.

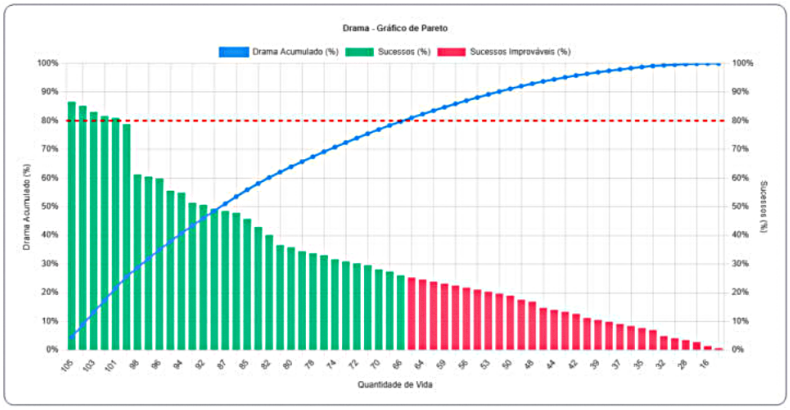

Figura 5. Resultados da métrica de Drama (Pareto 80/20).

O primeiro aspecto a ser discutido é a eficácia da plataforma como ferramenta de tradução de dados. Historicamente, designers de jogos lidam com imensas matrizes de telemetria bruta que exigem o uso de scripts de *Python* ou instâncias de *PowerBI* configuradas manualmente. A nossa aplicação simplificou este pipeline (ETL) abstraindo o processamento matemático pesado como o cálculo do Interquartil (IQR) para mitigação de *outliers* e as heurísticas espaciais do A^* para medir entropia procedural devolvendo ao desenvolvedor *insights* imediatos.

O achado mais provocativo reside na métrica de Equilíbrio. A esmagadora dominância da estratégia *Contar Passos* (+120% de variação) seria diagnosticada como uma falha crítica em um jogo *Multiplayer* ou *eSport*, configurando uma meta-estratégia inquebrável que asfixiaria a diversidade de *gameplay*. Entretanto, ao contextualizarmos o gênero do produto analisado (*Roguelike Singleplayer*), essa dominância é, na verdade, a comprovação do sucesso do *design*. Jogos dessa taxonomia são construídos em torno da "maestria sistêmica"[Demartini 2023]; espera-se que o jogador evolua da ignorância de mecânicas (*Sempre em Frente*) para o planejamento ótimo (*Contar Passos*). O balanceamento não exige que todas as heurísticas sejam viáveis, mas que a heurística baseada em raciocínio analítico seja amplamente recompensada, como os dados factualmente atestaram.

A Entropia Geral (18,25%) e a Incerteza (40,49%) validam a *Procedural Generation* do nível. Uma entropia excessiva (acima de 50%) indicaria um jogo pautado puramente em sorte (*RNG - Random Number Generator* mal calibrado), enquanto uma entropia nula o tornaria um *puzzle* determinístico. O equilíbrio encontrado mantém a tensão narrativa da *permadeath* sem subtrair o controle das mãos do usuário. A análise de Drama e Permanência fornece à equipe de *Level Design* um diagnóstico claro: o jogo opera de forma instável (lideranças curtas, em média de 3 fases) e torna-se criticamente dramático quando o *Health Point* cruza a barreira dos 65 pontos, dado que serve de gatilho paramétrico para que o designer ajuste os *drop-rates* de cura de acordo com a curva de dificuldade desejada.

Apesar do êxito na validação das métricas, a ferramenta apresentou limitações estruturais que afunilam sua adoção em escala. A dependência do módulo injetor de telemetria descontinuado [David e Castro 2019] restringiu os testes a projetos legados desenvolvidos estritamente na *engine* Unity. A arquitetura exige que o desenvolvedor capture o

arquivo JSON externamente (no Firebase) e realize o *upload* manual no portal, o que interrompe o fluxo contínuo de *Quality Assurance*. Ademais, por ter sido validada em um único jogo 2D estruturado em *grid*, as heurísticas de validação espacial (como o Algoritmo A^*) demandarão reescrita conceitual caso a ferramenta seja escalada para sistemas tridimensionais complexos (*NavMeshes*) ou jogos com mecânicas de liderança não vinculadas à barra de vida.

6. Conclusão

O presente estudo introduziu e validou a Flairplay Engine, uma plataforma conteinerizada de telemetria e balanceamento estruturada para o auxílio quantitativo na disciplina de *Game Design*. As hipóteses teóricas postuladas pela tese DRAGO foram transmutadas com sucesso para algoritmos funcionais, resultando em um sistema capaz de ler a estocástica de *gameplays* reais e compor painéis de visualização compreensíveis e factuais.

Foi demonstrado que a plataforma não tenta suplantar a decisão do *game designer*, atuando como um farol analítico. O estudo de caso com o *2D Roguelike* ilustrou como um desenvolvedor pode interpretar os resultados de Desequilíbrio e Entropia para ratificar a curva de aprendizado inerente a jogos modulares que adotam a "morte permanente", como vistos em títulos consagrados pela indústria [Demartini 2023]. A ferramenta revelou empiricamente que o jogo alvo encontra-se bem balanceado para o seu escopo, recompensando jogadores que decodificam os algoritmos de comportamento dos inimigos, garantindo um índice ideal de previsibilidade mesclado com sorte.

A superação do ceticismo quanto ao balanceamento matemático puro abre um considerável leque para investigações futuras. Trabalhos correlatos subsequentes poderão expandir o escopo da Flairplay Engine através da desintegração de sua camada de mensuração espacial, permitindo a ingestão de telemetria agnóstica de *engine* (Unreal, Godot, GameMaker) através de protocolos RESTful em tempo real via WebSockets.

Por fim, a integração de modelos de aprendizado de máquina (*Machine Learning*) para a clusterização não-supervisionada de estratégias poderia dispensar a anotação manual prévia exigida no sistema atual. Em vez de o testador registrar qual tática foi aplicada, a própria plataforma diagnosticaria padrões de movimentação e consumo de recursos, classificando o perfil psico-heurístico do jogador. Tais aprimoramentos têm o potencial de elevar a ferramenta ao patamar de padrão industrial, otimizando o ciclo de desenvolvimento de *software* interativo.

Referências

Adams, E. (2013). *Fundamentals of Game Design*. New Riders, Berkeley, CA, 3 edition.

Becker, A. e Görlich, D. (2020). What is game balancing? an examination of concepts. *ParadigmPlus*, 1(1).

Craddock, D. L. (2015). *Dungeon Hacks: How NetHack, Angband, and Other Roguelikes Changed the Course of Video Games*. Press Start Press.

David, A. M. e Castro, M. P. d. (2019). Uma abordagem para análise de qualidade de jogos baseada em telemetria: modelo e implementação na unity.

Demartini, F. (2023). O que é um jogo roguelike? Acesso em: 27 out. 2025.

Filho, F. R. F. et al. (2021). Balanceamento em jogos para dispositivos móveis: estudo de caso do jogo clash royale. In *Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames)*, pages 48–57, Gramado, RS.

Fonseca, M. D. D. (2024). Previsão de resultado de combates em dungeons and dragons 5ª edição.

Geloneze, F. R. e Arielo, F. S. (2017). Um breve análise sobre a indústria de jogos eletrônicos e os indie games. *Revista Multiplicidade*, 8(8).

Harris (2009). Column: @play: The berlin interpretation.

Jaffe, A. et al. (2012). Evaluating competitive game balance with restricted play. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 8(1).

Parreiras, M. V. C. (2025). *DRAGO - Design de Regras Avaliativas para Gameplays Otimizados*. PhD thesis, Universidade Federal do Rio de Janeiro.

Rashed, A. et al. (2025). A review of player engagement estimation in video games: Challenges and opportunities. *ACM Transactions on Multimedia Computing, Communications and Applications*, 21(7).

Ribeiro, C. D. (2018). Avaliação de algoritmos de detecção de colisão em jogos 2d para android. Master's thesis, Universidade Federal do Pampa (UNIPAMPA).

Salen, K. e Zimmerman, E. (2003). *Rules of Play: Game Design Fundamentals*. The MIT Press, Cambridge, MA.

Silva, M. A. R. d. (2018). *Pegasus: Uma Ferramenta de Simulação para Apoio ao Design de Jogos de Progressão*. PhD thesis, Universidade Federal do Rio de Janeiro.