Análise Formal de Gameplay com Redes de Petri Coloridas

Formal Gameplay Analysis Using Coloured Petri Nets

Hugo Gabriel Batista Borges[1], **Lucas Emerenciano Ramos**[1],
Joslaine Cristina Jeske de Freitas[1], **Críscilla Maia Costa Rezende**[1],
Franciny Medeiros Barreto[1]

[1]Instituto de Ciências Exatas e Tecnologia – Universidade Federal de Jataí (UFJ)
Caixa Postal 75801 - 615 – Jataí – GO – Brasil
{hugo.borges, lucas.ramos}@discente.ufj.edu.br
{joslaine, criscillarezende, franciny}@ufj.edu.br

Abstract. Introduction: *gameplay defines the player's actions and interactions when facing game challenges. Its analysis is useful for understanding game behavior and preventing project errors early in the design phases.* ***Objective:*** *to present a formal approach to analyze gameplay during game and level design, using Colored Petri Nets.* ***Methodology****: the game is modeled in CPNs, where activities become transitions, states become places, and the player is represented by a composite token carrying their progress. Formal verification requires defining key properties, such as event flow and the absence of deadlocks. Then, model simulation and property analysis are executed, based on the validation of the soundness property, to calculate all reachable markings.* ***Results:*** *verification allows for the automatic identification of design flaws and the simulation of multiple scenarios. The effectiveness of the approach was proven with examples from the game Detroit: Become Human.*
Keywords *Gameplay, Coloured Petri Nets, Formal Analysis, Detroit: Become Human, formal Method.*

Resumo. Introdução: *o gameplay define as ações e interações do jogador perante desafios do jogo. Sua análise é útil para entender o comportamento do jogo, evitar erros do projeto ainda nas fases de design.* ***Objetivo:*** *apresentar uma abordagem formal para analisar o gameplay durante o game e level design, utilizando Redes de Petri Coloridas.* ***Metodologia****: o jogo é modelado em CPNs, onde atividades viram transições, estados viram lugares e o jogador é representado por uma ficha composta com seu progresso. A verificação formal exige definir propriedades-chave, como fluxo de eventos e ausência de impasses. Depois, executa-se a simulação do modelo e a análise de propriedades, baseando-se na validação da propriedade soundness, para calcular todas as marcações alcançáveis.* ***Resultados:*** *a verificação permite identificar automaticamente falhas de design e simular múltiplos cenários. A eficácia da abordagem foi comprovada com exemplos do jogo Detroit: Become Human.*
Palavras-Chave *Gameplay, Redes de Petri Coloridas, Análise formal, Detroit: Become Human, Método formal*

1. Introdução

Um *videogame* é um meio de interação entre um jogador, uma máquina e, possivelmente, outros jogadores. Esse meio de interação é mediado por um contexto fictício formado na

mente do jogador por meio do reconhecimento e da interpretação de padrões visuais em uma tela eletrônica, bem como das relações entre esses padrões e as ações do jogador, por meio do qual o jogador atribui significado à sua experiência [Bergonse 2017].

O processo de criação de um *videogame* envolve duas etapas principais: *game* e *level design*. O *game design* é o processo de imaginar um jogo, definir como ele funciona, descrever seus elementos conceituais, funcionais, artísticos, dentre outros, e transmitir os detalhes do seu funcionamento pretendido para a equipe que irá desenvolvê-lo [Adams 2009].

Já o *level design* é a etapa onde os elementos definidos no *game design* se juntam para formar um ambiente imersivo, sendo definido como a criação de ambientes, cenários ou missões em um jogo eletrônico [Novak 2012].

Nas etapas de *game* e *level design* de um *videogame*, seu *gameplay* é definido [Guardiola 2019]. Esse conceito estabelece os desafios que um jogador deve enfrentar para alcançar um determinado objetivo de jogo e nas ações que o jogador pode realizar para lidar com esses desafios [Adams 2009]. Ele também define o grau e a natureza da interatividade que há no *videogame*, ou seja, como os jogadores podem interagir com o mundo do jogo e como esse mundo reage às escolhas que os jogadores fazem [Rouse III 2004].

Com o passar dos anos, os *videogames* têm apresentado *gameplays* cada vez mais complexos e interessantes. Essa evolução exige que os desenvolvedores assumam elevados riscos técnicos, pois há dificuldade em prever com precisão o comportamento de características inéditas do projeto, e seu impacto no sistema como um todo, adicionado aos riscos inerentes ao *game design*, como o nível de aceitação do público frente a novas funcionalidades [Blow 2004].

Esse aumento de complexidade destaca a importância de ferramentas para apoiar o desenvolvimento dos jogos em termos de tempo, custo e qualidade [Montero-Reyno e Carsí-Cubel 2009]. Sob a perspectiva de que a melhor maneira de capturar com precisão o comportamento de um sistema é produzir um modelo de seu comportamento em uma linguagem gráfica que possua semântica bem definida [Horrocks 1999].

Para esse propósito, cabem as Redes de Petri, ferramentas gráficas e matemáticas que oferecem um ambiente uniforme para modelagem, análise formal e projeto de sistemas de eventos discretos [Zhou e Zurawski 1995]. Uma das principais vantagens do uso de modelos de Redes de Petri é que o mesmo modelo pode ser utilizado tanto para a análise de propriedades comportamentais e avaliação de desempenho, quanto para a construção sistemática de simuladores e controladores de eventos discretos [van der Aalst e Stahl 2011].

Formalmente, uma Rede de Petri pode ser definida como uma quíntupla $PN = (P, T, F, W, M_0)$, na qual: P representa um conjunto finito de lugares; T é um conjunto finito de transições; e $F \subseteq (P \times T) \cup (T \times P)$ denota a relação de fluxo, ou seja, um conjunto de arcos que conectam lugares a transições e transições a lugares. A definição estabelece que os conjuntos de lugares e transições devem ser disjuntos ($P \cap T = \emptyset$) e que a união de ambos não pode ser um conjunto vazio ($P \cup T \neq \emptyset$). Além da estrutura básica, W atua como uma função de peso para os arcos, enquanto M_0 define a marcação

inicial da rede [Murata 1989].

As Redes de Petri possuem propriedades estruturais (independentes da marcação inicial) e comportamentais (dependentes da marcação inicial).Para a análise de sistemas, destacam-se as comportamentais conhecidas por boas propriedades, definidas da seguinte forma [Murata 1989, Cardoso e Valette 1997]: a **Alcançabilidade** (capacidade de atingir estados válidos a partir da marcação inicial); a **Limitabilidade** (garantia de que a quantidade de fichas nunca excede um limite finito); a **Vivacidade** (ausência total de *deadlocks*, assegurando que qualquer transição possa ser disparada futuramente); e a **Reversibilidade** (capacidade de retornar ao estado inicial ou a um estado de referência seguro). Adicionalmente, para modelos que representam fluxos de trabalho, é pertinente validar a propriedade ***Soundness***, que atesta a execução adequada do processo do início ao fim [van der Aalst e ter Hofstede 2000]. Essa propriedade é sustentada por três requisitos: **Alcançabilidade** da conclusão a partir de qualquer estado ativo; **Término Adequado**, exigindo que o processo finalize de forma limpa (com exatamente uma ficha no lugar final e nenhuma pendência na rede); e a **Ausência de Transições Mortas**, garantindo que todas as tarefas modeladas possuem a viabilidade real de serem executadas.

Porém, apesar de as Redes de Petri clássicas conseguirem modelar bem o comportamento de um sistema, em aplicações de larga escala os modelos tradicionais tendem a se tornar excessivamente extensos e de difícil compreensão, o que torna a modelagem um processo custoso. Além do problema de escalabilidade, as redes clássicas possuem um poder de expressão limitado, mostrando-se insuficientes para representar lógicas dinâmicas e regras condicionais complexas. Por fim, a incapacidade de modelar o tempo de maneira explícita impede a mensuração exata de quando os eventos ocorrem e de suas durações, o que dificulta a avaliação de desempenho do sistema modelado [van der Aalst e Stahl 2011].

Para suprir essas e outras limitações, foram criadas as Redes de Petri Coloridas (RPCs), que são redes de alto nível, pois representam uma linguagem gráfica e formal que combina o formalismo da teoria das Redes de Petri com linguagem de programação funcional [Jensen e Kristensen 2009].

Uma RPC é uma Rede de Petri na qual cada lugar tem um tipo, e cada ficha tem um valor (ou seja, uma cor) que corresponde ao tipo do lugar. Um arco em uma rede de Petri colorida pode ter uma inscrição de arco. Uma inscrição de arco é uma expressão com algumas variáveis que avaliam para um multiconjunto. Uma transição pode ter uma guarda. Uma guarda é uma condição e pode ter variáveis da mesma forma que as inscrições de arco possuem [van der Aalst e Stahl 2011].

Portanto, o objetivo desta pesquisa é propor uma abordagem formal para a análise do *gameplay* em *videogames* durante as fases de *game design* e *level design*. A abordagem proposta utilizou Redes de Petri Coloridas, pois oferecem alto poder expressivo para modelar cenários de jogo, e o software CPN Tools para a edição e simulação dos modelos desenvolvidos.

2. Trabalhos correlatos

Como discutido anteriormente, a crescente complexidade dos *videogames* requer abordagens mais rigorosas para mitigar riscos de desenvolvimento. Nesse contexto,

a literatura tem explorado a aplicação de métodos formais com Redes de Petri, em diferentes aspectos do *game design* e da simulação de sistemas interativos. A seguir, são apresentadas contribuições de diversos autores que propõem o uso desses métodos para a modelagem e análise de jogos.

Inicialmente, destaca-se o uso de Redes de Petri como ferramenta formal para modelar e simular comportamentos em jogos digitais na fase de pré-produção. O objetivo dessa abordagem é identificar falhas e otimizar decisões de *design* antes do desenvolvimento. Utilizando um jogo inspirado nas expedições portuguesas como estudo de caso, modelam-se duas frentes centrais do sistema: a interação com nativos por meio de uma rede de decisão baseada no contexto histórico e no tipo de solicitação e o comportamento de uma embarcação mercante, simulando variáveis como navegação, consumo de energia e tomada de decisão sob risco [Araújo e Roque 2009].

Sob outra perspectiva, uma abordagem conjunta foi desenvolvida para apoiar o processo de *game design* mediante a modelagem das atividades do jogador e do mapa topológico do mundo virtual. As atividades são representadas por redes de fluxo de trabalho (*WorkFlow nets*), enquanto os locais são descritos por grafos de estados. A interação entre os modelos ocorre via comunicação assíncrona, viabilizando a análise de propriedades comportamentais como a *soundness*, a qual garante a acessibilidade a todas as áreas e tarefas planejadas. A viabilidade do método é demonstrada com o jogo *Silent Hill II* e a ferramenta CPN Tools [Barreto e Julia 2021].

Em um contexto distinto, voltado aos esportes, as Redes de Petri também são aplicadas para modelar e analisar processos táticos em momentos decisivos de partidas da NBA. Tratando a partida como um sistema de eventos discretos, busca-se propor táticas ofensivas eficazes para situações críticas. Por meio do *software* PIPE, simulam-se jogadas como *killer tactic*, *isolation* e *pick-and-roll* para comparar tempos de execução e identificar as estratégias mais eficientes, fornecendo suporte analítico à tomada de decisão técnica [Liu e Wu 2013].

Por sua vez, nota-se também o desenvolvimento de metodologias focadas na identificação de falhas estruturais como *deadlocks*, *livelocks* e cenas inalcançáveis em jogos *multiplayer* baseados em cenas. O método gera automaticamente uma Rede de Petri Colorida a partir dos elementos do próprio jogo, mapeando de forma exaustiva as interações possíveis do usuário. A verificação formal ocorre por meio de ferramentas externas, permitindo a detecção de falhas de *design* sem a exigência de elaboração de um modelo paralelo construído do zero [Reuter et al. 2015].

Mais recentemente, a modelagem e a análise de jogos de quebra-cabeça utilizando Redes de Petri Coloridas foram exploradas com o intuito de mitigar o problema da explosão do espaço de estados. Tendo o jogo *Merchant Ship* como estudo de caso, a pesquisa integra a linguagem ML para definir funções complexas capazes de reduzir a redundância de estados. Essa estratégia viabiliza a extração de métricas de complexidade essenciais ao *game design*, tais como caminhos mínimos para o sucesso e taxas de falha [Taghinezhad-Niar e Pashazadeh 2024].

Os estudos citados propuseram métodos que fortalecem e contribuem fortemente quanto ao estado da arte de métodos formais e principalmente de aplicações utilizando Redes de Petri como meio. Entretanto, a análise do *gameplay* não foi o foco desses

trabalhos. Uma vez que o *gameplay* pode ser visto como o que acontece entre o início e o final de um jogo, enfatizando a necessidade de considerar uma forma de analisá-lo ainda nas fases de *game* e *level design*, a fim de representar a interação do jogador e identificar comportamentos e potenciais problemas. É nessa lacuna que esta pesquisa se insere, buscando modelar a interação do jogador a fim de identificar comportamentos sistêmicos e antever potenciais problemas de projeto.

3. Modelagem e Análise do Gameplay com Redes de Petri Coloridas

A análise formal do *gameplay* exige a transposição de mecânicas abstratas para modelos verificáveis. Nesse sentido, esta pesquisa propõe uma metodologia de modelagem baseada em Redes de Petri Coloridas. Para ilustrar a aplicação de suas etapas, utiliza-se como estudo de caso um trecho do capítulo "The Hostage" do jogo *Detroit: Become Human*, notório por suas alternativas narrativas. No cenário escolhido, o jogador assume um detetive androide que investiga uma cena de duplo homicídio para coletar pistas, preparando-se estrategicamente para confrontar um androide que emula emoções humanas e resgatar uma refém.

3.1. Definir os elementos do cenário de jogo

A primeira etapa é definir os elementos do cenário de jogo. As atividades consistem de ações que o jogador precisa executar. O jogador poderá executar essas ações de forma sequencial, paralela, iterativa ou escolher entre duas ou mais atividades. Ao realizar uma atividade qualquer, espera-se que o jogador mude de um estado de jogo para o outro.

Por exemplo, para o trecho do capítulo "The Hostage" que será utilizado como estudo de caso, a atividade é **procurar pistas em uma cena de crime**.

3.2. Definir o papel do jogador:

A segunda etapa consiste em definir o papel do jogador, considerando as atividades previamente definidas. Neste momento, é preciso detalhar, o que exatamente o jogador deverá desempenhar para concluir as atividades, a ordem dessas atividades, se são alternativas, sequenciais, iterativas, entre outras estruturas de fluxo. O objetivo dessa etapa é manter a especificação em um nível de detalhe suficiente para que o jogo possa ser "executado" no modelo proposto.

Por exemplo, os papéis que o jogador pode assumir no trecho de jogo em questão são:

$\Rightarrow$ Procurar por pistas: neste trecho o jogador interage livremente com o cenário, o que altera dinamicamente um medidor de sucesso (iniciado em 48%) e estabelece interações "conectadas" com o destravamento de eventos futuros.

- Investigar caixa de arma (+3% de sucesso)
- Ouvir ao fone de ouvido (+3% de sucesso)
- Analisar sangue de Daniel (+3% de sucesso)
- Examinar o sapato de Emma (+3% de sucesso)
- Aprender o nome do divergente (+7% de sucesso, conectado)
- Investigar o corpo do pai (+3% de sucesso):
  - Entender a causa do incidente (+7% de sucesso, conectado)

  - Não entender a causa do incidente
- Investigar o corpo do policial (+3% de sucesso):
  - Pegar a arma do policial (+7% de sucesso, conectado)
  - Deixar a arma do policial
- Não investigar a cena do crime

3.3. Criar o modelo formal

A terceira etapa consiste na construção do modelo formal, transpondo as especificações do cenário e do jogador para as Redes de Petri Coloridas. Nessa estrutura, os lugares representam os estados do jogo, as transições denotam as atividades e os arcos definem o fluxo de atividades. O principal diferencial da RPC para a modelagem de jogos é a capacidade de associar dados às fichas [Barreto et al. 2014].

Dessa forma, configura-se a ficha do jogador convertendo o estado do personagem (como a posse de itens) em variáveis estruturadas, como booleanos ou inteiros. Essa representação de dados é essencial para conferir precisão à análise formal, garantindo que o modelo transcenda um mero fluxograma. O comportamento do sistema é orquestrado nas transições, ações com pré-requisitos são condicionadas por funções de guarda, enquanto as consequências dessas ações atualizam dinamicamente os atributos da ficha por meio de expressões nos arcos de saída. A Figura 1 ilustra o modelo resultante correspondente às atividades mapeadas na etapa anterior.

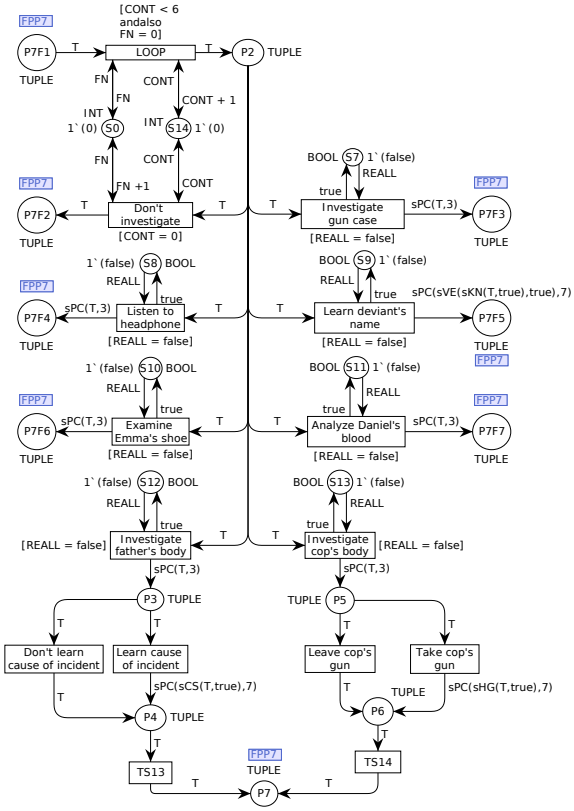

Figura 1. Modelo da investigação da cena do crime.

Nesta parte do capítulo, a mecânica do jogo permite ao jogador interagir com

múltiplos elementos, optar por não interagir com nada ou realizar mais de uma atividade, sob a restrição de que cada atividade só pode ser executada uma única vez.

Para representar esse comportamento no modelo, foi implementada uma estrutura de repetição juntamente com um mecanismo de controle na disponibilidade das transições. Esse controle garante que, uma vez disparada, a transição correspondente àquela atividade não possa ser executada novamente. É possível observar esses mecanismos nos lugares prefixados com "S" acima das transições apresentadas no modelo.

Para modelar o cenário do jogo *Detroit: Become Human*, o jogador é representado por uma ficha principal que trafega pelo modelo. O estado e o progresso do personagem são armazenados nessa ficha, que pertence a um conjunto de cores (*colset*) customizado do tipo `TUPLE`.

A marcação inicial da ficha é definida por `1'(player, 48, false, false, false, false, false)`, compreendendo sete variáveis estruturadas em *colsets* específicos. Dentre elas, a variável **`P`** (`PLAYER`) representa o próprio jogador, enquanto **`PC`** (`PERAC`) mensura a probabilidade de sucesso da missão, iniciada em 48%. As demais variáveis funcionam como sinalizadores lógicos: **`HG`** (`HELDGUN`) indica se o jogador possui uma arma, **`IC`** (`ISCLOSE`) indica se o jogador se aproximou do divergente; **`CS`** (`CAUSE`) aponta se a causa do incidente foi identificada, **`VE`** (`VOCEMMA`) indica sobre a relação entre o divergente e a refém, e, por fim, **`KN`** (`KNOWNAME`) sinaliza se o nome do divergente foi descoberto.

A dinâmica do modelo é regida por funções de atribuição (do tipo *setter*) inseridas nas inscrições dos arcos. Essas funções operam sobre a ficha para atualizar os estados do jogo, por exemplo, a instrução `sPC(T, 3)` incrementa o valor de `PC` em três unidades. Esse mecanismo juntamente com o encapsulamento da ficha em `T` elimina a necessidade de reescrever a ficha completa para alterar os seus valores.

3.4. Estabelecer aspectos do gameplay para análise

Apesar da diversidade de gêneros de jogos dificultar a criação de métricas universais, a literatura identifica três aspectos estruturais comuns a serem avaliados em diferentes *gameplays*, como o **fluxo de eventos** (a relação de causa e efeito entre as ações do jogador e as respostas do sistema), a **alcançabilidade de estados** (a garantia de que todas as possibilidades do jogo podem ser exploradas livremente, sem restrições indevidas) e a **ausência de impasses** (a prevenção contra falhas de especificação que causem travamentos irreversíveis na progressão) [Lankoski e Björk 2015].

3.5. Análise formal

A análise formal do modelo no *software* CPN Tools é conduzida sob duas abordagens complementares: a simulação passo a passo e a análise de propriedades do modelo através cálculo do Espaço de Estados (*State Space*).

3.5.1. Simulação passo a passo

A simulação manual viabiliza a verificação visual da dinâmica do jogo, permitindo rastrear as alterações lógicas nas variáveis da ficha a cada disparo de transição. O desfecho

da simulação passo a passo para o cenário investigado é ilustrado na Figura 2.

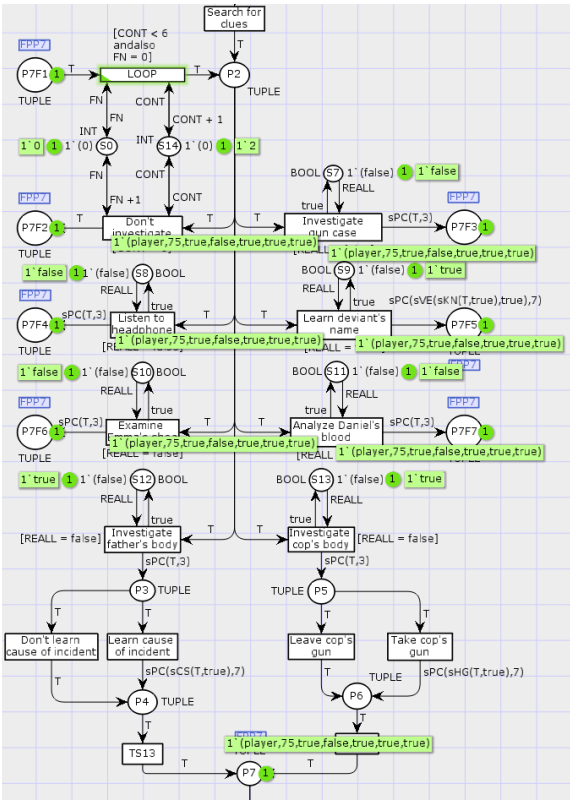

Figura 2. Disparo das transições *Investigate cop's body* e *Take cop's gun*.

A análise da marcação alcançada no lugar `P7` evidencia a consolidação dos dados na ficha do jogador ao final da investigação. Os atributos atualizados indicam que o personagem obteve êxito em aprender o nome do divergente (`KN=true`), descobrir sua relação prévia com a refém (`VE=true`), compreender a causa do incidente (`CS=true`) e adquirir a arma do policial (`HG=true`). O somatório dessas interações estratégicas resultou em um incremento acumulado de 27% na probabilidade de sucesso (`PC`), preparando o detetive adequadamente para o confronto iminente.

3.5.2. Análise das propriedades

A análise com espaço de Estados mapeia um grafo com todas as marcações alcançáveis para validar propriedades comportamentais de forma automática. Contudo, a variação excessiva dos dados da ficha principal durante o modelo do *gameplay* pode gerar a explosão do espaço de estados [Jensen 1996]. Para solucionar esse gargalo matemático e possibilitar a verificação da propriedade de *soundness*, o modelo de análise exige três adaptações estruturais, como a **redução das instâncias** da ficha, minimizando suas variações de valor sem perder a semântica original, a **readequação das funções de guarda**, ajustando os pré-requisitos para operarem com a ficha simplificada, e a inclusão de uma **estrutura de reinicialização**, adicionando uma transição *t** que conecta o lugar final ao inicial para fechar o ciclo do sistema e validar a prova matemática [van der Aalst e ter Hofstede 2000, Barreto e Julia 2021]. Modelos de jogos podem ser

interpretados como fluxos de trabalho, dada a natureza de suas atividades, que seguem sequências, alternativas e/ou ciclos, o que torna válida a utilização da propriedade de *soundness* para atestar a correção do fluxo de atividades do jogo.

Na Figura 3 pode ser observado o modelo modificado seguindo os critérios especificados para a análise da propriedade *soundness*, e na Figura 4 o relatório do espaço de estados resultante do CPN tools.

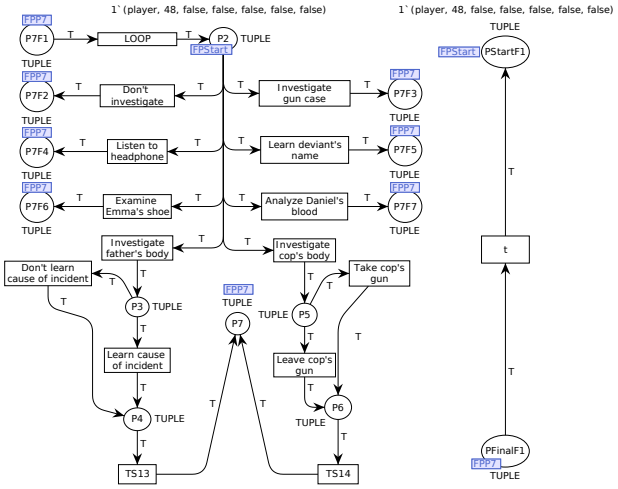

Figura 3. Modelo modificado para a análise da propriedade *Soundness*.

Statistics

State Space
Nodes: 6
Arcs: 16
Secs: 0
Status: Full

Scc Graph
Nodes: 1
Arcs: 0
Secs: 0

Boundedness Properties

Best Integer Bounds

	Upper	Lower
[All nodes]	1	1	0

Best Upper Multi-set Bounds
[All nodes] 1
1`(player,48,false,false,false,false,false)

Best Lower Multi-set Bounds
[All nodes] 1 empty

Home Properties

Home Markings
All

Liveness Properties

Dead Markings
None

Dead Transition Instances
None

Live Transition Instances
All

Fairness Properties

Impartial Transition Instances
None

Fair Transition Instances
DBH_model'TS13 1
DBH_model'TS14 1

Just Transition Instances
DBH_model'Analyze_Daniel's_blood 1
DBH_model'Don't_investigate 1
DBH_model'Don't_learn_cause_of_incident 1
DBH_model'Examine_Emma's_shoe 1
DBH_model'Investigate_cop's_body 1
DBH_model'Investigate_father's_body 1
DBH_model'Investigate_gun_case 1
DBH_model'LOOP 1
DBH_model'Learn_cause_of_incident 1
DBH_model'Learn_deviant's_name 1
DBH_model'Leave_cop's_gun 1
DBH_model'Listen_to_headphone 1
DBH_model'Take_cop's_gun 1
DBH_model't 1

Transition Instances with No Fairness
None

Figura 4. Relatório *State Space* do modelo apresentado na Figura 3.

O relatório do Espaço de Estados Figura 4 demonstra que o modelo satisfaz as propriedades de vivacidade (*liveness*) e limitabilidade (*boundedness*), atestando que a

rede não modificada é *sound*. A avaliação do *gameplay* extraída dessa análise baseia-se em três pilares: fluxo de eventos, alcançabilidade e ausência de impasses. O fluxo de eventos é mapeado por um grafo de 6 nós e 16 arcos, representando exaustivamente todas as rotas e decisões possíveis do jogador. O *status* completo do relatório confirma que todas as marcações são alcançáveis, e a análise do Grafo SCC (*Strongly Connected Component*), que resultou em um único nó, prova que os 6 estados estão fortemente conectados. Por fim, a ausência de transições e marcações mortas (*dead transitions* e *dead markings*) assegura que o sistema é totalmente livre de *deadlocks*. No contexto do jogo, isso garante matematicamente que o jogador jamais enfrentará travamentos irreversíveis ou bloqueios de progressão decorrentes de falhas *game design*, especificamente, em termos de *gameplay*.

4. Conclusão e resultados

Esta pesquisa propôs e validou uma abordagem formal para a análise de *gameplay* durante as fases de *game* e *level design*, utilizando Redes de Petri Coloridas. A principal contribuição deste estudo reside na entrega de um método que ao traduzir regras e mecânicas contidas em um cenário de jogo para um formalismo matemático, possibilita a identificação e correção de falhas estruturais do *gameplay* antes de codificar uma linha sequer.

A aplicação da metodologia no trecho inicial do capítulo *"The Hostage"* de *Detroit: Become Human* evidenciou a capacidade das Redes de Petri Coloridas em modelar *gameplays* complexos e narrativas ramificadas. A eficácia do método foi confirmada tanto pela simulação passo a passo, que validou a evolução funcional dos estados, quanto pela análise de propriedades (fluxo, alcançabilidade, ausência de impasses e *Soundness*), que atestou a integridade estrutural do modelo. Conclui-se, portanto, que a abordagem estabelece um paralelo preciso entre o comportamento do jogador e as estruturas da rede, destacando-se pela capacidade de atrelar múltiplos detalhes interdependentes.

Cabe ressaltar que o uso do jogo *Detroit: Become Human* neste trabalho serviu apenas para exemplificar a abordagem proposta. Contudo, a abordagem apresentada neste trabalho é recomendada para as etapas iniciais de desenvolvimento, com o intuito de prevenir falhas de *game design*. Com sua adoção, seria possível definir o *gameplay*, fluxo e atividades do jogador, dispensando qualquer esforço de codificação. Dessa forma, a validação do *gameplay* pode ocorrer por meio de um modelo de simulação gráfico e formal, que servirá de apoio para etapas posteriores do processo de desenvolvimento do jogo.

Como trabalhos futuros, seria interessante aplicar o método proposto em um projeto efetivo de jogo que ainda está sendo construído, logo nas etapas iniciais de desenvolvimento. Assim, seria possível evidenciar as consequências projetuais da análise, validar a transição entre o modelo verificado e a decisão de design por meio da aplicação em um caso real. Essa aproximação ajudará a fortalecer, na prática, a relação do método formal proposto com os processos de *game* e *level design*.

Referências

Adams, E. (2009). *Fundamentals of Game Design*. New Riders Publishing, USA, 2nd edition.

Araújo, M. e Roque, L. (2009). Modeling games with petri nets. In *Proceedings of the 2009 DiGRA International Conference: Breaking New Ground: Innovation in Games, Play, Practice and Theory (DiGRA 2009)*, London, UK. DiGRA.

Barreto, F. M., de Freitas, J. C. J., Soares, M. S., e Julia, S. (2014). A straightforward introduction to formal methods using coloured petri nets. In *International Conference on Enterprise Information Systems*, volume 2, pages 145–152. SCITEPRESS.

Barreto, F. M. e Julia, S. (2021). Formal approach based on petri nets for modeling and verification of video games. *Computing and Informatics*, 40(1):216–248.

Bergonse, R. (2017). Fifty years on, what exactly is a videogame? an essentialistic definitional approach. *The Computer Games Journal*, 6(4):239–255.

Blow, J. (2004). Game development: Harder than you think: Ten or twenty years ago it was all fun and games. now it's blood, sweat, and code. *Queue*, 1(10):28–37.

Cardoso, J. e Valette, R. (1997). *Redes de petri*. Editora da UFSC, Florianópolis, SC.

Guardiola, E. (2019). Gameplay definition: A game design perspective. *Proceeding of the Game On 19 conference*, pages 5–10.

Horrocks, I. (1999). *Constructing the User Interface with Statecharts*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

Jensen, K. (1996). An introduction to the practical use of coloured petri nets. *Advanced Course on Petri Nets*, 1(1):237–292.

Jensen, K. e Kristensen, L. M. (2009). *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*. Springer, Berlin, Heidelberg, 1 edition.

Lankoski, P. e Björk, S. (2015). Formal analysis of gameplay. In Lankoski, P. e Björk, S., editors, *Game Research Methods: An Overview*, pages 23–35. ETC Press, Pittsburgh, PA.

Liu, Y. e Wu, W. (2013). Petri net modeling analysis of processes at clutch time in nba games. In *2013 IEEE International Conference of IEEE Region 10 (TENCON 2013)*, pages 1–4, Piscataway, NJ. IEEE.

Montero-Reyno, E. e Carsí-Cubel, J. Á. (2009). A platform-independent model for videogame gameplay specification. In *Proceedings of DiGRA 2009 Conference: Breaking New Ground: Innovation in Games, Play, Practice and Theory*, New York, NY, USA. Association for Computing Machinery.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *vol*, 77:541–580.

Novak, J. (2012). *Game Development Essentials: An Introduction*. Delmar, Cengage Learning, Boston, Massachusetts, EUA, 3 edition.

Reuter, C., Göbel, S., e Steinmetz, R. (2015). Detecting structural errors in scene-based multiplayer games using automatically generated petri nets. In *Proceedings of the 10th*

International Conference on the Foundations of Digital Games (FDG 2015), pages 1–9, Pacific Grove, CA, USA. Society for the Advancement of the Science of Digital Games. Conference held June 22-25, 2015. Copyright held by author(s).

Rouse III, R. (2004). *Game Design Theory and Practice*. Wordware Publishing Inc., Plano, TX, USA, 2 edition.

Taghinezhad-Niar, A. e Pashazadeh, S. (2024). State-space analysis and complexity assessment of puzzle games using colored petri nets. *International Journal of Web Research*, 7(4):13–27.

van der Aalst, W. e Stahl, C. (2011). *Modeling Business Processes: A Petri Net-Oriented Approach*. The MIT Press, Cambridge, MA.

van der Aalst, W. M. P. e ter Hofstede, A. H. M. (2000). Verification of workflow task structures: A petri-net-based approach. *Information Systems*, 25(1):43–69.

Zhou, M. e Zurawski, R. (1995). *Introduction to Petri Nets in Flexible and Agile Automation*. Petri Nets in Flexible and Agile Automation. Kluwer Academic Publishers, Amsterdam, Netherlands.