**UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO**
**PRÓ-REITORIA DE GRADUAÇÃO**
**DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIA**
**CURSO INTERDISCIPLINAR EM TECNOLOGIA DA INFORMAÇÃO**

**PEDRO VINÍCIUS DE ANDRADE QUEIROZ**

# DESENVOLVIMENTO DE UM SISTEMA ESPECIALISTA COM MOTOR DE INFERÊNCIA UTILIZANDO PROLOG PARA O JOGO CLUE SUSPEITOS®

**PAU DOS FERROS**
**2025**

**PEDRO VINÍCIUS DE ANDRADE QUEIROZ**

# DESENVOLVIMENTO DE UM SISTEMA ESPECIALISTA COM MOTOR DE INFERÊNCIA UTILIZANDO PROLOG PARA O JOGO CLUE SUSPEITOS®

Monografia apresentada à Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Interdisciplinar em Tecnologia da Informação.

Orientador: Prof. Dr. Kennedy Reurison Lopes

**PAU DOS FERROS**

**2025**

Q3d Queiroz, Pedro Vinícius de Andrade.
Desenvolvimento de um sistema especialista com motor de inferência utilizando Prolog para o jogo Clue Suspeitos® / Pedro Vinícius de Andrade Queiroz. - 2025.
51 f. : il.

Orientador: Kennedy Reurison Lopes.
Monografia (graduação) - Universidade Federal Rural do Semi-árido, Curso de Tecnologia da Informação, 2025.

1. Sistema Especialista. 2. Prolog. 3. Clue Suspeitos®. 4. Inteligência Artificial. I. Lopes, Kennedy Reurison, orient. II. Título.

Ficha catalográfica elaborada por sistema gerador automático em conformidade com AACR2 e os dados fornecidos pelo) autor(a).
Biblioteca Campus Pau dos Ferros
Bibliotecária: Erlanda Maria Lopes da Silva
CRB: 15/966

PEDRO VINÍCIUS DE ANDRADE QUEIROZ

DESENVOLVIMENTO DE UM SISTEMA ESPECIALISTA COM MOTOR DE INFERÊNCIA UTILIZANDO PROLOG PARA O JOGO CLUE SUSPEITOS®

Monografia apresentada à Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Interdisciplinar em Tecnologia da Informação.

Defendida em: 11 de Dezembro de 2025

BANCA EXAMINADORA

Kennedy Reurison Lopes, Prof. Dr. (UFERSA)
Presidente

Bruno Francisco Xavier, Prof. Dr. (UFERSA)
Membro Examinador

João Batista de Souza Neto, Prof. Dr. (UFERSA)
Membro Examinador

## DEDICATÓRIA

Dedico esse trabalho a Deus e a todos os amigos e familiares que me ajudaram ao longo dessa jornada.

## AGRADECIMENTOS

Em primeiro lugar, agradeço a Deus, por todas as bênçãos e oportunidades que recebi, e por me conceder a força necessária para enfrentar cada desafio desta caminhada.

Agradeço profundamente à minha família, em especial aos meus pais, Ana Maria e Francisco Zildalécio, pelo esforço contínuo e pelos sacrifícios realizados para que eu pudesse completar esta etapa; ao meu irmão Paulo, à minha madrinha Valda, à minha prima Ana Clara, à minha tia Fátima, ao meu tio Walber e aos meus avós Valfredo, Lurdes, Nascimento e Paula. A todos os membros da minha família, cujo apoio e incentivo foram fundamentais para a realização dos meus estudos, meu sincero reconhecimento.

Aos meus amigos, que sempre representaram um pilar de apoio ao longo do curso, Pollyana, Pedro Hércules, Davi, Michely, Matheus, Vitor, Ângela, Andrey, Neilla, João Felype, Nathan, Renan e tantos outros que não citei, mas que levo comigo em meu coração, meu muito obrigado.

Expresso minha imensa gratidão ao meu professor orientador, Dr. Kennedy Reurison, cuja paciência, dedicação e orientação foram essenciais para o desenvolvimento deste trabalho.

Agradeço também a Sebastiana Jaqueline pelo apoio e auxílio incansáveis ao longo destes anos. Sua presença foi indispensável para que eu chegasse até aqui.

Por fim, agradeço a todos que, de alguma forma, contribuíram para que eu pudesse alcançar este objetivo.

## EPÍGRAFE

"Não se esqueça de sorrir em qualquer situação. Enquanto estiver vivo, coisas boas virão, e serão muitas."

(Eiichiro Oda)

## RESUMO

Este trabalho apresenta o desenvolvimento de um Sistema Especialista (SE) com Motor de Inferência em Prolog para o jogo Clue Suspeitos®. O objetivo foi modelar o raciocínio lógico-dedutivo necessário para identificar o autor do crime, o local e a arma, por meio de uma base de conhecimento estruturada em regras lógicas. O processo envolveu a compreensão de Sistemas Especialistas, a aplicação de Inteligência Artificial em jogos estratégicos e a formalização das interações do jogo em decisões automatizadas. O núcleo do SE consistiu na criação de regras que orientam as quatro operações fundamentais do jogo: anotações, perguntas, respostas e acusações, simulando o raciocínio de um especialista humano. O motor de inferência atualiza dinamicamente o estado do jogo, prioriza deduções relevantes e fornece orientações consistentes ao usuário. Sua implementação em Prolog garantiu formalização clara e execução eficiente das inferências. A validação por meio de cenários de teste demonstrou que o sistema realiza deduções corretas, mantém desempenho estável e oferece explicações coerentes. O trabalho contribui para a área de sistemas aplicados a jogos, fornecendo um modelo estruturado de inferência lógica que pode servir como referência para futuras pesquisas e aplicações em ambientes de simulação estratégica.



**Palavras-chave:** sistema especialista; prolog; clue suspeitos®; inteligência artificial.

## ABSTRACT

This paper presents the development of an Expert System (ES) with an Inference Engine in Prolog for the game Clue Suspects Card Game®. The objective was to model the logical-deductive reasoning necessary to identify the perpetrator of the crime, the location, and the weapon, using a knowledge base structured around logical rules. The process involved understanding Expert Systems, applying Artificial Intelligence to strategic games, and formalizing game interactions into automated decisions.The core of the ES consisted of creating rules that guide the four fundamental operations of the game: notes, questions, answers, and accusations, simulating the reasoning of a human expert. The inference engine dynamically updates the game state, prioritizes relevant deductions, and provides consistent guidance to the user. Its implementation in Prolog ensured clear formalization and efficient execution of inferences.Validation through test scenarios demonstrated that the system makes correct deductions, maintains stable performance, and offers coherent explanations. The work contributes to the field of game-applied systems, providing a structured model of logical inference that can serve as a reference for future research and applications in strategic simulation environments.



**Keywords:** expert system; prolog; clue suspects card game®; artificial intelligence.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| IA | *Inteligência Artificial* |
| NPC | *Non-Player Character* |
| SE | Sistema Especialista |

**SUMÁRIO**

# 1 INTRODUÇÃO

A evolução tecnológica torna-se palco de transformações significativas em diversas áreas, destacando-se especialmente, nas últimas décadas, a área de *Inteligência Artificial* (IA) que tem se consolidado progressivamente como uma das principais tecnologias presentes nas nossas ações do cotidiano e até em aplicações diversas e específicas, podendo variar desde assistentes virtuais a softwares capazes de realizar tarefas complexas com exatidão e precisão (Russell *et al.*, 2010).

Nesse contexto, os Sistemas Especialistas (SEs) surgem como uma das primeiras e mais tradicionais abordagens da IA, cujo objetivo é capturar e aplicar o conhecimento especializado para resolver problemas complexos que exigem raciocínio semelhante ao humano. Por meio da formalização de regras e fatos, esses sistemas permitem a automação de decisões em domínios específicos, tornando possível a aplicação da IA em áreas onde o conhecimento especializado é fundamental.

De acordo com Barreto e Prezoto (2010), os SEs operam por meio da atribuição de relações entre informações previamente coletadas em entrevistas detalhadas conduzidas entre o engenheiro de conhecimento e o especialista da área. Essa abordagem de interpretação das informações faz com que sistemas desse tipo se adaptem bem a determinados tipos de problemas, como em jogos de raciocínio lógico.

Surgido no mesmo período em que os SEs foram desenvolvidos, o Prolog é uma linguagem de programação lógica utilizada para implementar motores de inferência baseados na lógica de primeira ordem, que é o seu principal paradigma (Bratko, 2001). Levando em consideração os SEs e a linguagem Prolog, o objeto de estudo escolhido para este trabalho é o jogo Clue Suspeitos®, inspirado no clássico Clue e em jogos do gênero Detetive. Sua premissa se adapta bem ao formato lógico e estruturado exigido por uma base de conhecimento. Como descrito nas instruções oficiais do Clue Suspeitos (HASBRO, 2018), o objetivo dos jogadores é descobrir a configuração do crime – composta por uma arma, um local e um suspeito – que foi aleatoriamente definida no início da partida. A dinâmica de informações ocultas e o objetivo de desvendar o crime antes dos demais jogadores – formulando perguntas e ocultando dados estratégicos – cria um cenário competitivo pouco previsível, mas com um grande número de variáveis a serem analisadas, tornando o jogo especialmente apropriado para a aplicação de SEs.

## 1.1 Objetivos

O presente Trabalho de Conclusão de Curso (TCC) tem como objetivo desenvolver um SE com um motor de inferência lógica, utilizando a linguagem Prolog, aplicado ao jogo *Clue Suspeitos*®. O sistema buscará representar, de forma automatizada, o raciocínio dedutivo necessário para identificar o autor do crime, o local e a arma, por meio de uma base de conhecimento e um conjunto estruturado de regras lógicas.

A proposta fundamenta-se nos princípios dos SEs, visando explorar a lógica declarativa como abordagem para tomadas de decisão nas deduções típicas do jogo. O foco está na construção de um sistema que una a modelagem lógica das regras do jogo à implementação de inferências automatizadas em Prolog. Como objetivos específicos, têm-se:

- Elaborar uma base de dados a partir da modelagem das regras do jogo.
- Desenvolver um sistema capaz de interagir com o usuário do SE.
- Validar o desempenho do SE por meio de testes com diferentes cenários do jogo, analisando a consistência e a eficiência das inferências realizadas.
- Comparar os resultados por meio de simulações observáveis das decisões lógicas efetuadas.
- Apresentar as explicações dedutivas tomadas pelo motor de inferência.
- Documentar o processo de desenvolvimento, destacando os desafios enfrentados e os aprendizados obtidos na construção do SE.

## 1.2 Organização do Trabalho

Este trabalho está organizado em seis capítulos. Primeiramente, o Capítulo 2 aborda a fundamentação teórica necessária para a compreensão dos conceitos que embasam o desenvolvimento do estudo. Nesse sentido, no Capítulo 3 são apresentados os trabalhos que abordam a criação de softwares semelhantes, evidenciando sua função e aplicabilidade no contexto dos jogos. Posteriormente, o Capítulo 4 apresenta a metodologia aplicada para alcançar os objetivos traçados. No Capítulo 5 são apresentados os resultados da construção do SE, abordando desde a formulação e execução de testes até a validação do funcionamento do sistema. Por fim, no Capítulo 6 são discutidas as conclusões e os possíveis trabalhos futuros.

# 2 FUNDAMENTAÇÃO TEÓRICA

Os conceitos que fundamentam o desenvolvimento deste trabalho serão expostos neste capítulo, no qual são discutidas teorias e princípios essenciais para a compreensão completa do estudo. A estruturação do capítulo está dividida em quatro seções: a seção 2.1 introduz os conceitos de Inteligência Artificial, contextualizando as abordagens que levaram ao surgimento dos SEs; a seção 2.2 introduz os SEs como modelos de IA capazes de realizar inferências lógicas baseadas em conhecimento; a seção 2.3 aborda o Prolog como linguagem de programação lógica capaz de desenvolver motores de inferência para SEs; e, por fim, a seção 2.4 apresenta o jogo Clue Suspeitos® como cenário de aplicação para o estudo dos processos de dedução lógica.

## 2.1 Inteligência Artificial

A Inteligência Artificial (IA) é o campo da ciência e engenharia que estuda a criação de agentes inteligentes, capazes de perceber informações do ambiente em que estão e tomar ações com base nelas (Russell *et al.*, 2010). A busca por esse objetivo levou à organização do campo da IA em duas dimensões principais: a primeira delas mede o sucesso com base no desempenho humano ou segundo um padrão ideal de racionalidade, enquanto a segunda distingue entre pensamento e ação (Russell *et al.*, 2010). A combinação dessas duas dimensões resulta em quatro abordagens distintas:

- **Sistemas que agem como seres humanos**: Inspirados pelo Teste de Turing, refere-se a sistemas capazes de produzir respostas indistinguíveis das de um ser humano. Diante de uma interação, um avaliador não conseguiria identificar se está dialogando com uma máquina ou com outro humano. Já existem modelos que superaram parcialmente o teste de Turing (Jones; Bergen, 2025). Em uma pesquisa conduzida por Cameron R. Jones e Benjamin K. Bergen do Departamento de Ciência Cognitiva da Universidade da Califórnia em San Diego, foram avaliados quatro sistemas: ELIZA, GPT-4o, LLaMa-3.1-405B, e GPT-4.5. Dentre eles, destacou-se o GPT-4.5, capaz de enganar 73% dos participantes, maior índice de sucesso dentre os sistemas avaliados.
- **Sistemas que pensam como seres humanos**: São sistemas construídos para simular processos cognitivos semelhantes aos de um ser humano. Essa abordagem integra teorias da psicologia e da neurociência, buscando modelos que contribuam para a compreensão do funcionamento da mente humana.

- **Sistemas que agem racionalmente**: Nessa categoria são desenvolvidos agentes racionais, sistemas orientados a agir da melhor forma possível com base nas informações disponíveis, maximizando resultados esperados. Essa abordagem é mais conveniente ao desenvolvimento científico do que as técnicas de pensamento e de agir humano, por possuir racionalidade bem definida, genérica e reutilizável.
- **Sistemas que pensam racionalmente**: Baseiam-se na lógica formal desenvolvida desde Aristóteles, com foco na representação e manipulação de pensamentos estruturados. Esse método considera que a mente funciona através de leis de pensamento, padrões de argumentos que permitem alcançar conclusões corretas ao serem alimentados com premissas corretas. Essa vertente inspirou lógicos do século XIX a criar notações simbólicas que serviriam de base para os primeiros sistemas inteligentes.

## 2.2 Sistemas Especialistas

Originados na época da Segunda Guerra Mundial, os Sistemas Especialistas (SEs) são um dos primeiros ramos de estudo da Inteligência Artificial, sendo responsáveis por marcar a transição entre desenvolvimento de software de propósito geral e programas voltados a objetivos e domínios específicos. Esse tipo de sistema tem como objetivo o desenvolvimento de sistemas capazes de tomar decisões ou realizar ações de forma similar a um especialista humano em um domínio específico de conhecimento (Negnevitsky, 2005).

Como será mais detalhado posteriormente, a base de funcionamento dos sistemas especialistas é o conhecimento condensado obtido diretamente de especialistas, descrito por Costa e Silva (2005) como alguém que possui conhecimento aprofundado em um determinado domínio de aplicação. Vale destacar que embora um SE bem desenvolvido possa, de fato, realizar operações concisas e objetivas, existem tarefas nas quais eles não conseguem substituir especialistas humanos, o que torna sua aplicação mais comum em softwares de apoio à decisão. Entre suas principais aplicações, destacam-se: apoio a diagnósticos médicos, nos quais o sistema pode operar com dados clínicos (como histórico do paciente e sintomas) para prever possíveis doenças; suporte empresarial, atuando na análise de dados de mercado para auxiliar decisões estratégicas; jogos de raciocínio lógico, em que o sistema deduz jogadas otimizadas a partir do estado atual do jogo; além de diversas outras áreas onde seja viável extrair conhecimento estruturado a partir da experiência de especialistas.

Os SEs baseados em regras de produção constituem o modelo mais clássico dessa

categoria de sistemas, sendo amplamente utilizados por representar o conhecimento de forma explícita através de regras do tipo “SE-ENTÃO” Buchanan e Smith (1988). Como explica Negnevitsky (2005), o pensamento humano não pode ser facilmente representado por algoritmos devido à sua natureza interna e complexa, o que torna necessário expressar esse conhecimento por meio de regras de produção. Essas regras seguem o formato “SE” (premissa ou condição) “ENTÃO” (conclusão ou ação), onde, se a primeira parte for verdadeira, a segunda será executada. Por exemplo:

```
SE tenho dinheiro
ENTÃO posso ir à praia
```

O exemplo apresentado pode ser interpretado como uma regra de produção simples, expressa no formato “SE–ENTÃO”. A parte “SE” indica a condição que deve ser satisfeita para que a parte “ENTÃO” execute uma ação ou gere uma conclusão. Assim, se a condição “tenho dinheiro” for verdadeira, chega-se à conclusão de que “posso ir à praia”. Formalmente, essa regra estabelece uma relação condicional entre uma premissa e uma consequência, permitindo que o sistema deduza automaticamente a ação apropriada para quando a condição é atendida (Negnevitsky, 2005). Além do formato básico “SE–ENTÃO”, as regras de produção podem utilizar conectivos lógicos para combinar múltiplas condições ou permitir múltiplas alternativas. Os conectivos mais comuns são “E” (conjunção) e “OU” (disjunção). Observe o uso do conectivo “E”, utilizado quando todas as condições devem ser satisfeitas para que a ação seja executada:

```
SE tenho dinheiro
E está ensolarado
ENTÃO posso ir à praia
```

Neste caso, tanto a condição “tenho dinheiro” quanto “está ensolarado” precisam ser verdadeiras para que a conclusão “posso ir à praia” seja acionada. Já o conectivo “OU” é empregado quando basta que apenas uma das condições seja verdadeira para que a conclusão seja acionada. Por exemplo:

```
SE tenho dinheiro
OU recebi um convite
ENTÃO posso ir à praia
```

Aqui, tanto “tenho dinheiro” como “recebi um convite” podem satisfazer a conclusão desde que pelo menos uma delas seja verdadeira.

É no formato de regras de produção que os SEs baseados em regras armazenam o conhecimento advindo do especialista humano em sua base de conhecimento, um dos três principais componentes desse tipo de sistema. Essa base reúne as informações relativas ao domínio do problema, que são extraídas do especialista por meio de entrevistas conduzidas por um Engenheiro de Conhecimento (Buchanan; Smith, 1988).

A interface com o usuário é responsável por realizar a comunicação entre o usuário final e a máquina, permitindo a entrada de solicitações pelo usuário e exibindo os resultados obtidos através do processo realizado pelo motor de inferência (Barreto; Prezoto, 2010).

O motor de inferência, por sua vez, é responsável por interpretar as regras existentes na base de conhecimento e aplicar mecanismos lógicos para inferir conclusões a partir das informações disponibilizadas Barreto e Prezoto (2010). Esse componente é fundamental para o raciocínio automatizado do sistema. Neste trabalho, o motor de inferência é desenvolvido em Prolog, como será detalhado mais à frente.

Além dos três módulos principais, os SEs podem contar com módulos complementares que acrescentam novas funcionalidades, ainda que não sejam estritamente necessários para o funcionamento do sistema. Um exemplo relevante é o módulo de explicação, que funciona como uma subcamada entre o motor de inferência e a interface com o usuário. Como explica (Negnevitsky, 2005), seu papel é detalhar o processo de inferência, permitindo ao usuário visualizar o raciocínio do sistema e as regras utilizadas para chegar a determinada conclusão. Esse recurso é útil para aumentar a confiabilidade do sistema e facilitar sua validação por parte dos usuários, especialmente em tarefas em que não é possível substituir completamente um especialista humano, como para diagnósticos médicos (Costa; Silva, 2005).

Outro exemplo de componente que pode ser de grande utilidade em alguns casos é a memória de trabalho, anotações registradas no formato de regras “SE-ENTÃO” que acrescentam informações extras sobre um caso e que podem ser empregadas para realização de inferências personalizadas.

Os SEs baseados em regras apresentam uma série de vantagens e benefícios que merecem ser levados em consideração. Um dos principais pontos positivos é a clareza na representação do conhecimento. As regras do tipo “SE-ENTÃO” podem ser facilmente compreendidas tanto por desenvolvedores quanto por especialistas, o que facilita o desenvolvimento e validação do sistema.

Além disso, esse tipo de abordagem adota uma estrutura uniforme, onde cada regra representa uma unidade independente de conhecimento, permitindo que seja modular e auto-documentável. Ela também permite a separação entre o conhecimento e seu processamento: a divisão entre motor de inferência e a base de conhecimento favorece a reutilização, a escalabilidade do sistema (Negnevitsky, 2005).

Também é importante discorrer acerca das desvantagens características dessa estrutura. Embora cada regra possa ser facilmente compreendida de maneira independente, uma grande quantidade de regras pode tornar o sistema difícil de compreender, principalmente pela ausência de uma hierarquia clara que defina o papel e a importância relativa de cada regra dentro do conjunto. Outro ponto crítico é o tempo de execução em sistemas que possuem um número elevado de regras, o que pode comprometer o desempenho e tornar inviável seu uso em aplicações em tempo real. Além disso, os SEs baseados em regras não são capazes de aprender automaticamente com base na experiência do usuário, diferente de especialistas humanos, que conseguem adaptar seu raciocínio e reconhecer cenários que necessitam seguir caminhos fora do comum. Logo, esses sistemas dependem da constante atualização manual por parte de um especialista humano (Negnevitsky, 2005).

Neste trabalho, a abordagem baseada em regras foi escolhida por sua robustez e compatibilidade com o problema, permitindo formular fatos e regras sobre o jogo de maneira estruturada e flexível. Apesar de algumas limitações, os Sistemas Especialistas baseados em regras mostraram-se adequados para atender aos objetivos deste projeto, especialmente por se adequarem à natureza determinística presente no jogo Clue Suspeitos.

## 2.3 Prolog

Prolog é uma linguagem de programação lógica baseada em predicados de primeira ordem. Diferente de linguagens imperativas como C ou Java, em que instruções são executadas de maneira sequencial, o Prolog utiliza a formulação de problemas lógicos, que são resolvidos por um mecanismo de inferência. Essa característica torna a linguagem adequada para diversas aplicações que envolvem raciocínio simbólico e manipulação de conhecimento. Entre elas se destacam: a inteligência artificial, com agentes capazes de tomar decisões a partir de regras lógicas; os sistemas especialistas, utilizados em diagnósticos, configuração de produtos e suporte à decisão; e o processamento de linguagem natural, que inclui análise sintática e semântica, interpretação de texto e compreensão da linguagem humana (Clocksin; Mellish, 2003).

Como descrito por Bratko (2001), a capacidade do Prolog se dá em função de três características principais:

- Unificação: É o processo pelo qual o Prolog tenta tornar dois termos iguais, associando variáveis a valores ou verificando a correspondência entre termos. A unificação permite que consultas sejam comparadas com fatos ou regras, possibilitando que o mecanismo de inferência descubra quais valores satisfazem as condições de um predicado.
- Encadeamento regressivo (backward chaining): Esse mecanismo trabalha de maneira inversa, começando pela conclusão desejada e procurando premissas ou fatos que a satisfaçam. Em outras palavras, o Prolog tenta provar o objetivo verificando se ele pode ser derivado a partir de regras e fatos existentes.
- Backtracking: É o mecanismo de busca que permite ao Prolog explorar alternativas. Caso uma tentativa de unificação ou uma regra falhe, o sistema retorna a um ponto anterior da árvore de busca e tenta outras possibilidades até encontrar uma solução ou esgotar todas as alternativas. Isso possibilita a exploração completa de todas as soluções possíveis para uma consulta.

A programação em Prolog é estruturada a partir da declaração de fatos e da definição de regras (Clocksin; Mellish, 2003). Em sistemas baseados em lógica, um fato representa uma informação elementar sobre o domínio do problema, representando relações entre objetos. Cada fato é composto por termos, que podem ser átomos (constantes simbólicas), números (valores literais inteiros ou reais) ou variáveis (que podem assumir diferentes valores durante a execução). Existem também variáveis anônimas (_), que indicam que o valor do termo não é relevante para a inferência.

Sua estrutura padrão segue o formato:

```
predicado(arg1, arg2, ..., argN).
```

O predicado descreve uma relação ou propriedade, enquanto os argumentos (também chamados de termos) especificam os elementos envolvidos nessa relação (Clocksin; Mellish, 2003). Por exemplo, o conjunto de fatos:

```
pai(lucas, clara).
pai(lucas, roberto).
pai(roberto, joao).
```

Os fatos nomeados `pai/2`, indicam uma relação de parentesco entre dois indivíduos que foram declarados como argumentos. Analisando esse exemplo conseguimos inferir que Lucas possui relação paternal com Clara e Roberto, que por sua vez possui com João. Nesse caso, os termos lucas, clara, roberto e joao são átomos, ou seja, constantes simbólicas que representam indivíduos específicos. Além disso, a declaração utilizada, `pai/2`, apresenta duas características fundamentais do Prolog: o functor, que corresponde ao nome do predicado (pai) e indica a relação ou a propriedade que está sendo representada, e a aridade, representada pelo número após a barra (/2), que indica que o predicado possui dois argumentos, ou seja, dois termos que participam da relação definida pelo functor.

Conforme explica Bratko (2001) as regras representam relações que não estão explicitamente declaradas como fatos, mas que podem ser inferidas a partir deles. Cada regra define uma conclusão possível quando determinadas condições são satisfeitas. A estrutura da regra é formada por cabeçalho e corpo. O corpo pode combinar múltiplas condições usando conectivos lógicos, como os operadores de conjunção e disjunção, apresentados na Tabela 1.

Tabela 1 – Principais operadores lógicos utilizados em Prolog

| **Operador** | **Significado** | **Exemplo** | **Descrição** |
|---|---|---|---|
| `:-` | Implicação (SE–ENTÃO) | `C :- A, B.` | Define uma regra: C é verdadeiro quando A e B forem verdadeiros. |
| `,` | Conjunção (E) | `A, B` | Verdadeiro quando A **e** B forem verdadeiros; usado para encadear condições. |
| `;` | Disjunção (OU) | `A ; B` | Verdadeiro quando A **ou** B forem verdadeiros; representa escolha lógica. |
| `->` | If–Then | `A -> B` | Executa B se A for verdadeiro; usado principalmente em construções de controle. |
| `-> ;` | If–Then–Else | `A -> B ; C` | Se A for verdadeiro, executa B; caso contrário, executa C. |
| `\+` | Negação | `\+ A` | É verdadeiro caso não seja possível provar que A é verdadeiro. |
| `=` | Unificação | `X = Y` | X e Y podem ser unificados; estabelece igualdade lógica entre termos. |
| `\==` | Não unificação | `X \== Y` | Verdadeiro quando X e Y **não** puderem ser unificados. |

**Fonte**: Autoria Própria (2025)

Retomando o exemplo anterior, podemos criar regras capazes de inferir graus de parentesco sobre os indivíduos apresentados:

```
eh_avo(X, joao) :-
    pai(X, Y),
    pai(Y, joao).
```

Esta regra é composta por duas partes: o cabeçalho, representado pelo trecho `eh_avo(X, joao) :-`, que indica que estamos inferindo quem é avô de João, e o corpo, que contém as condições que devem ser satisfeitas para que a conclusão seja verdadeira, representadas por `pai(X, Y), pai(Y, joao)`. A variável **X** representa o candidato a avô, podendo ser inserida como um nome pessoal, por exemplo. Nessa regra é possível identificar elementos centrais da sintaxe, como as variáveis **X** e **Y** (identificadas por iniciarem com letra maiúscula), átomos (uma constante, como o caso do termo **joao**) e o operador lógico E, representado pela vírgula (**,**) que aparece entre os fatos (Clocksin; Mellish, 2003). Com base nesse conjunto de fatos, ainda é possível inferir outros graus de parentesco, como tios, filhos e netos.

Segundo Merritt (2012), a estrutura das regras em Prolog se assemelha à das regras de produção "SE–ENTÃO". A principal diferença está na ordem declarativa: em Prolog, a conclusão (equivalente ao "ENTÃO") aparece antes do corpo da regra (equivalente ao "SE"). Dessa forma, a lógica de inferência permanece a mesma, apenas adaptada à sintaxe da linguagem.

Para a implementação dos exemplos apresentados, assim como deste projeto, utilizou-se o SWI-Prolog, um dos interpretadores de Prolog mais difundidos na comunidade acadêmica e de desenvolvimento. Ele oferece um ambiente interativo que permite a definição de fatos, regras e consultas de maneira direta, além de disponibilizar recursos como depuração e bibliotecas para manipulação de listas e arquivos (Wielemaker, 2003). A escolha do SWI-Prolog se deu em função de sua fácil integração com Python, por meio da biblioteca PySWIP, possibilitando a manipulação e leitura de arquivos em Prolog por meio de chamadas de queries no Python.

## 2.4 O Jogo Clue Suspeitos®

O jogo Clue Suspeitos® é inspirado no clássico Clue e em jogos do gênero Detetive. O objetivo do jogo é descobrir os elementos do crime: arma do crime, local e suspeito. Essas características são representadas por um grupo de 15 cartas, sendo quatro (4) armas, cinco (5) lugares e seis (6) suspeitos. Ao início de cada partida, uma carta de cada grupo é removida de

maneira aleatória, formando um crime e restando 12 cartas que são embaralhadas e distribuídas igualmente aos jogadores (de 2 a 4 jogadores por partida), compondo as evidências de cada jogador (HASBRO, 2018).

Para alcançar o objetivo de descobrir o cenário do crime, o jogo possui quatro mecânicas principais:

- **Perguntar:** ação realizada por um jogador ao jogador seguinte (geralmente o que está à esquerda), questionando sobre cartas de tipos diferentes, como arma e lugar, lugar e suspeito, ou suspeito e arma. Caso o próximo jogador não possua resposta para a pergunta, esta deve ser feita ao jogador posterior a ele, e assim por diante até percorrer todos os jogadores.
- **Responder:** o jogador que for questionado deve responder de forma honesta, de acordo com suas cartas de evidência (distribuídas antes do início da partida). Caso a pergunta acerte apenas uma de suas cartas, o jogador deve exibi-la ao questionador, sem que os demais jogadores vejam seu conteúdo. Se possuir ambas as cartas mencionadas, ele poderá escolher qual mostrar, sem necessariamente revelar que possui as duas. Caso não tenha nenhuma das cartas questionadas, deve responder: "Não posso te ajudar."Vale ressaltar que o processo de pergunta e resposta é público para os demais jogadores, que sabem quem perguntou, para quem, quais cartas foram mencionadas e se houve resposta ou não.
- **Anotar:** além das cartas de evidência, cada jogador recebe 15 cartas contendo os mesmos elementos das cartas do crime e das evidências. Essas cartas são utilizadas como ferramenta de anotação, auxiliando os jogadores a registrar e organizar as informações obtidas ao longo da partida.
- **Acusar:** é o momento final da partida, no qual os jogadores utilizam as informações coletadas e suas anotações para formular um palpite sobre o crime. Em cada turno, o jogador pode escolher entre perguntar ou acusar. No caso de uma acusação, todos os jogadores devem montar um trio de cartas com a suposição sobre o crime e colocá-lo virado para baixo na mesa. Em seguida, revelam-se as cartas do crime e os palpites de cada jogador. Vence aquele que acertar primeiro, a partir do jogador que iniciou a acusação.

Observando esse conjunto de mecânicas de jogo, é possível elaborar um conjunto de fatos e regras capazes de analisar o cenário atual da partida e retornar as melhores escolhas que podem

ser tomadas pelo jogador. Para complementar essa modelagem, foram considerados também cenários ilustrativos que representam situações recorrentes durante uma partida e que influenciam diretamente as estratégias de dedução adotadas pelos jogadores. Esses cenários permitem demonstrar como o Sistema Especialista interpreta evidências parciais, atualiza hipóteses e restringe o espaço de busca.

Situação 1: Pergunta voltada para a confirmação de suspeita sobre um tipo específico de carta. Nessa situação, o Jogador A formula ao Jogador B a seguinte pergunta: "Você tem Cozinha ou Faca?". A intenção estratégica por trás dessa pergunta é verificar especificamente Cozinha, pois o Jogador A já possui boas pistas de que local pode ser o verdadeiro local do crime. A carta Faca é incluída apenas como elemento auxiliar, com o propósito de obrigar o oponente a responder.

A interpretação das respostas possíveis segue a lógica abaixo:

- Se o Jogador B responder "Cozinha": Elimina essa possibilidade como local do crime.
- Se o Jogador B responder "Faca": Elimina essa arma nas anotações. Essa resposta ainda não anula a chance de Cozinha ser o local do crime, necessitando que o Jogador A realize outra pergunta envolvendo Cozinha e uma carta adicional restrita.
- Se o Jogador B responder "Não Posso te Ajudar": Os jogadores da mesa, incluindo o Jogador A, descobrem que Jogador B não possui a carta. Para o Jogador A isso se torna um forte indício de que Cozinha compõe a cena do crime.

Situação 2: Coleta de informações públicas em perguntas externas. O Jogador C pergunta ao Jogador D: "Você tem Peacock ou Sala de Estar?". Mesmo que a pergunta seja feita em público, a resposta sobre a carta é privada, expondo apenas a posse ou não de uma das duas cartas. O objetivo estratégico aqui envolve compreender essa resposta como uma informação extra. Observemos as respostas:

- Se o Jogador D realizar uma resposta negativa: Neste caso, todos os jogadores agora sabem que o Jogador D não possui nenhuma das cartas, aumentando a chance de que elas componham o cenário do crime. Imaginando também o cenário em que D possui ambas as cartas, a resposta escolhida pode influenciar positiva ou negativamente nas escolhas realizadas por C.
- Se o Jogador D realizar uma resposta positiva: O jogador que recebe a resposta direta (Jogador C) elimina a carta de suas anotações, impactando em sua própria estratégia. Os demais jogadores, Jogador A e Jogador B, agora sabem que D possui pelo menos

uma das cartas. Diferentes cenários podem se originar dessa resposta, dentre eles: caso A ou B possuam uma das cartas da pergunta, ou saibam que outro jogador a possui, eles podem realizar o descarte da outra; caso as cartas constem em outras perguntas com resposta negativa, isso indica que elas estão envolvidas no crime; e dentre outras conclusões, incluindo fatores como erros (perguntas repetidas) ou blefes (o jogador que fez a pergunta possui alguma das cartas perguntadas ou sabe que o próximo jogador não as possui).

Situação 3: Eliminação de cartas por fatores de incerteza. Sabendo que o Jogador C possui Faca ou Spa com base em uma resposta anterior, o Jogador A opta por removê-las de suas perguntas ou acusações até que possua mais informações. Seu objetivo é maximizar a chance de acerto de um palpite ou obtenção de uma resposta, ao mesmo tempo que explora a distribuição de evidências na partida.

Essas situações evidenciam tanto a complexidade técnica do jogo Clue Suspeitos quanto o funcionamento das decisões lógicas tomadas pelos jogadores durante uma partida. Essa análise ressalta que a proposta deste trabalho, o desenvolvimento de um Sistema Especialista, é adequada para essa tarefa.

# 3 TRABALHOS RELACIONADOS

Neste capítulo são apresentados os principais estudos e referências que fundamentam o desenvolvimento do sistema especialista proposto. Inicialmente, são discutidas as aplicações de sistemas especialistas em diferentes contextos e domínios, destacando como essas soluções podem resolver problemas complexos por meio de regras e inferências. Em seguida, é abordado o papel da lógica e da linguagem Prolog no contexto da Inteligência Artificial. Por fim, são exploradas as diversas formas de aplicação da Inteligência Artificial no campo dos jogos.

## 3.1 Aplicações de Sistemas Especialistas

Os Sistemas Especialistas (SEs) representam um dos campos mais antigos e consolidados da Inteligência Artificial, tendo como marcos históricos projetos como o Dendral e o Mycin, que demonstram o valor de transformar conhecimento especialista em regras lógicas. O **Dendral**, criado por Edward Feigenbaum, Georgia Sutherland e Bruce Buchanan na década de 1960, utilizava heurísticas e algoritmos de geração e teste com o objetivo de auxiliar químicos na determinação de estruturas moleculares por meio da análise de dados de espectrometria de massa (Buchanan; Sutherland; Feigenbaum, 1969). Já o **Mycin**, desenvolvido por Edward Shortliffe na década de 1970, empregava um motor de inferência baseado em regras de produção associado a fatores de certeza para realizar diagnósticos e recomendar tratamentos contra infecções bacterianas, oferecendo suporte à decisão clínica de forma consistente e transparente (Shortliffe, 2012).

Apesar de terem surgido há mais de meio século, os SEs ainda continuam a desempenhar um papel bastante relevante em diversos domínios, especialmente em áreas que exigem decisões objetivas, consistentes e explicáveis. Essa continuidade do paradigma pode ser observada em sistemas modernos que aplicam motores de inferência e bases de conhecimento estruturadas em contextos clínicos, administrativos e industriais. Por exemplo, de forma semelhante ao que o Mycin já fazia na década de 1970, o **PROTUV (Protocolo para Tratamento de Úlceras Venosas)** é um sistema especialista desenvolvido para apoiar a tomada de decisão dos enfermeiros na terapia tópica de úlceras venosas. Ele sistematiza a aplicação de protocolos clínicos, facilita o registro e o acompanhamento da evolução do tratamento e dos custos envolvidos, além de sugerir a conduta tópica mais adequada com base nas características da lesão (Sellmer *et al.*, 2013).

A estrutura do PROTUV segue o modelo clássico de sistemas especialistas, bastante semelhante àquela empregada neste trabalho, sendo composta por três elementos principais: a

base de conhecimento, motor de inferência e interface. A base de conhecimento armazena regras de produção com informações sobre o procedimento de tratamento das úlceras. O motor de inferência foi implementado utilizando a ferramenta Drools, que cruza os dados inseridos pelo enfermeiro com as regras da base de conhecimento para fornecer apoio à tomada de decisão. Por fim, a interface permite o registro das úlceras, acompanhando o tratamento, indicando recomendações personalizadas, gerando relatórios e acessando um glossário online para auxiliar na interpretação de procedimentos ou produtos desconhecidos pelo usuário.

Embora a maior parte das aplicações modernas se concentre em saúde, indústria e sistemas corporativos, as características dos SEs também se mostram adequadas para ambientes que envolvem dedução lógica. Jogos de investigação, como Clue Suspeitos, dependem de elementos empregados em SEs, como formulação de hipóteses, eliminação sistemática de possibilidades e uso de evidências para inferir conclusões. Nesse contexto, o desenvolvimento de um SE para apoiar decisões ou simular comportamento inteligente em jogos desse tipo representa uma aplicação direta e ainda pouco explorada desse paradigma.

## 3.2 Lógica e Prolog na Inteligência Artificial

A lógica é um dos pilares da IA, permitindo representar conhecimento de forma explícita e realizar inferências a partir de regras definidas. Ela fornece mecanismos formais de dedução de novas informações, sendo especialmente importante em domínios que exigem decisões consistentes e explicáveis (Colmerauer; Roussel, 1996).

O Prolog, desenvolvido por Alain Colmerauer e Philippe Roussel, é uma linguagem clássica voltada para raciocínio lógico e Inteligência Artificial Simbólica (IAS). Ela utiliza fatos, regras e mecanismos de inferência baseados em unificação e encadeamento, oferecendo uma estrutura declarativa que separa o conhecimento do motor de inferência (Bratko, 2001). Clocksin e Mellish (2003) detalham como essa abordagem permite implementar SEs, agentes inteligentes e outras aplicações que dependem de dedução lógica automática de informações.

Um exemplo contemporâneo do uso de Prolog para desenvolvimento de sistemas é o **Sistema Pericial PROLOG Convencional Capacitado (PROLOG$^{cc}$)**, desenvolvido por Siqueira e Rocio no trabalho “Sistema Pericial em Prolog no Diagnóstico do Potencial Agrícola de Solos”, que apresenta uma aplicação de programação lógica combinada com raciocínio difuso para avaliar deficiências físicas, químicas e nutricionais do solo (Siqueira; Rocio, 2011). O sistema realiza inferências por meio de regras difusas do tipo SE–ENTÃO, que integram os

dados laboratoriais fuzzificados a um conjunto de regras projetado com base no conhecimento de especialistas. Após o processo dedutivo, uma etapa de defuzzificação gera um valor percentual que indica o nível de deficiência do solo e o esforço necessário para sua requalificação. O estudo demonstrou a viabilidade do uso de Prolog em conjunto com lógica difusa para diagnósticos agrícolas, reforçando o papel da representação lógica e dos motores dedutivos em aplicações que exigem interpretação transparente sob condições de incerteza.

Um exemplo de aplicação contemporânea de Prolog em jogos é o ***Non-Player Character Decision-Making With Prolog and Ontologies*** apresentado por Lapeyrade e Rey (2023), que combina regras de Prolog e ontologias de domínio para modelar características, preferências e objetivos de *Non-Player Character*s (NPCs) em uma versão mais complexa do Wumpus World implementada no motor Unity (Lapeyrade; Rey, 2023). A abordagem permite inferir objetivos, determinar ações possíveis e selecionar a ação mais adequada com base em utilidade, utilizando ontologias de personalidade e de ações.

Esse método apresenta uma alternativa aos mecanismos tradicionais, como árvores de comportamento ou máquinas de estados, oferecendo maior flexibilidade, reutilização do conhecimento e facilidade de manutenção em jogos complexos. A arquitetura do sistema integra o Unity como ambiente de jogo com o SWI-Prolog como motor de inferência, comunicando-se via interface de socket para atualizar o estado do jogo, executar o raciocínio lógico e retornar a ação escolhida. Esse exemplo evidencia como Prolog continua sendo uma ferramenta poderosa para implementar IA simbólica em jogos, permitindo raciocínio explícito, decisões consistentes e modularidade na criação de agentes inteligentes.

No presente trabalho, Prolog é utilizado como motor de inferência do sistema especialista para o jogo Clue Suspeitos. Para isso, assim como no trabalho de Lapeyrade e Rey, emprega-se o SWI-Prolog, aqui integrado com Python através da biblioteca PySWIP, permitindo que hipóteses sobre suspeitos, armas e locais sejam representadas como fatos e regras. O sistema avalia essas hipóteses de forma lógica, eliminando possibilidades inconsistentes e deduzindo conclusões com base nas evidências do jogo. Essa aplicação evidencia a relevância contínua do paradigma dos SEs e da lógica simbólica em contextos modernos, além de mostrar a viabilidade de integração entre linguagens de programação e motores de inferência em projetos contemporâneos.

## 3.3 Inteligência Artificial em Jogos

O uso de Inteligência Artificial em jogos desempenhou um papel fundamental no desenvolvimento da área, servindo como laboratório para testar técnicas de busca, tomada de decisão, representação de conhecimento e aprendizado de máquina. Jogos oferecem ambientes controlados, com regras claras e objetivos bem definidos, permitindo avaliar algoritmos de forma concreta e mensurável. Por esse motivo, desde os primórdios da IA, eles se tornaram cenários amplamente utilizados para demonstração de capacidades computacionais que simulam raciocínio estratégico e adaptabilidade. Esse campo não apenas impulsionou avanços científicos, mas também influenciou diretamente o estudo de agentes inteligentes em aplicações reais, como robótica e sistemas de planejamento automatizado.

Nesse contexto, o trabalho de Arthur Samuel, ***Some Studies in Machine Learning Using the Game of Checkers*** (Samuel, 1959), é amplamente reconhecido como uma das primeiras aplicações bem-sucedidas de aprendizado de máquina. Ele empregou técnicas como minimax, heurísticas avaliativas e aprendizado por reforço, criando um agente capaz de melhorar sua performance ao competir repetidamente contra si mesmo. Esse estudo estabeleceu conceitos fundamentais para sistemas baseados em experiência, demonstrando que jogos podem ser utilizados para validar estratégias de autoaperfeiçoamento computacional.

Décadas depois, a evolução da IA em jogos alcançou um novo patamar com o trabalho de Feng-Hsiung Hsu, ***IBM's Deep Blue: Chess Grandmaster Chips*** (Hsu, 1999). O estudo descreve a criação de hardware especializado para acelerar buscas em árvores de decisão no xadrez, permitindo processar milhões de posições por segundo por meio de variantes otimizadas do algoritmo minimax. Essa abordagem culminou nos avanços que levaram o sistema Deep Blue a derrotar o campeão mundial de xadrez Garry Kasparov em 1997, demonstrando a potência de arquiteturas híbridas que combinam heurísticas especializadas, modelos determinísticos de busca e processamento paralelo em larga escala.

Trazendo para a atualidade, o trabalho de Silver *et al.* (2016), ***Mastering the game of Go with deep neural networks and tree search***, combina redes neurais profundas com o algoritmo de Monte Carlo Tree Search para jogar Go, um jogo complexo devido à enorme quantidade de estados possíveis e à dificuldade de avaliar posições intermediárias. A arquitetura integra duas redes neurais principais: a Rede de Política, que estima a probabilidade de cada movimento promissor e reduz a largura de busca, e a Rede de Valor, que estima o resultado provável de uma posição e diminui a necessidade de simulações profundas. Essas redes trabalham em conjunto

com uma versão aprimorada do algoritmo de Monte Carlo Tree Search. Esse modelo resultou em um sistema capaz de superar todos os programas concorrentes e derrotar jogadores profissionais, demonstrando o impacto da integração entre aprendizado profundo e algoritmos clássicos de busca na construção de agentes inteligentes altamente competitivos.

De modo geral, os trabalhos apresentados demonstram como o desenvolvimento de IA em jogos serviu tanto para impulsionar novas técnicas quanto para consolidar paradigmas já existentes. Desde agentes baseados em heurísticas e busca, como visto nos estudos de Samuel, passando por arquiteturas híbridas de alto desempenho como o Deep Blue, até sistemas apoiados em modelos estatísticos complexos, como o AlphaGo, observa-se uma progressão contínua na complexidade e na sofisticação das técnicas utilizadas, sempre se adaptando à natureza do problema. Nesse contexto, os sistemas simbólicos permanecem relevantes por sua capacidade de representar conhecimento de forma explícita e interpretável. Assim, a utilização de Prolog e de princípios de sistemas especialistas neste trabalho se alinha a essa tradição, evidenciando que mecanismos de inferência lógica continuam adequados para modelar processos decisórios estruturados em ambientes de jogo, complementando abordagens estatísticas contemporâneas.

# 4 METODOLOGIA

As decisões metodológicas adotadas neste trabalho foram estruturadas de modo a permitir, por meio de processos e procedimentos, a obtenção dos resultados estabelecidos tanto no objetivo geral quanto nos objetivos específicos apresentados na seção 1.1. A construção do SE para o jogo Clue Suspeitos exigiu a combinação de diferentes abordagens metodológicas, garantindo embasamento teórico, implementação prática do sistema e análise de seu funcionamento em cenários simulados.

## 4.1 Pesquisa e Abordagem Metodológica

A pesquisa possui caráter misto, reunindo elementos quantitativos, experimentais e documentais. A abordagem qualitativa está presente na análise das mecânicas do jogo Clue Suspeitos e na formulação das regras que compõem a base de conhecimento, uma vez que o processo de dedução não depende de dados estatísticos, mas sim da interpretação e estruturação do conhecimento adquirido durante as partidas. O componente documental se fundamenta no levantamento de materiais teóricos, artigos científicos, livros e dissertações, que embasam os conceitos de SEs e programação lógica utilizados ao longo do projeto. Por fim, o aspecto experimental se manifesta na execução do protótipo em cenários de jogo simulados, permitindo observar o comportamento do motor de inferência diante de diferentes estados de jogo e verificar se as respostas geradas seguem o raciocínio esperado.

Paralelamente à definição da abordagem metodológica, foi realizado um levantamento bibliográfico que fundamenta os conceitos aplicados ao sistema. Nessa etapa, foram consultados diversos autores que contribuem para áreas como Sistemas Especialistas, Inteligência Artificial e desenvolvimento em Prolog. Esses conhecimentos foram adquiridos a partir de dissertações, artigos científicos e livros. Sendo assim, os autores teóricos que fundamentam a proposta são os seguintes: Barreto e Prezoto (2010), Bratko (2001), Buchanan, Sutherland e Feigenbaum (1969), Clocksin e Mellish (2003), Costa e Silva (2005), Gomes (2010), HASBRO (2018), Jones e Bergen (2025), Koehler (1998), McCarthy *et al.* (2006), Mendes (1997), Merritt (2012), Negnevitsky (2005), Oliveira e Silva (2013), Ribeiro *et al.* (2019), Russell *et al.* (2010). Essas referências fornecem a base conceitual e as técnicas necessárias para compreender as características dos SEs baseados em regras, o funcionamento do motor de inferência e os fundamentos da linguagem Prolog.

## 4.2 Modelagem do Conhecimento

Com a fundamentação teórica estabelecida, iniciou-se a modelagem do conhecimento responsável por estruturar formalmente o domínio do jogo. Essa etapa consistiu em traduzir a lógica utilizada para uma base de fatos e regras capaz de sustentar o processo de inferência. Fatos foram empregados para representar elementos estáticos do jogo, como cartas, evidências e descartes, enquanto regras expressaram os mecanismos de dedução que descrevem como novas informações podem ser inferidas a partir das ações dos jogadores.

A construção dessa base tomou como referência tanto as regras oficiais quanto o comportamento observado em partidas simuladas. A análise do ganho de informações em diferentes situações de pergunta e resposta e as diferentes estratégias adotadas pelos jogadores serviram de base para ajustar a lógica do sistema e garantir que este reproduzisse corretamente o processo de raciocínio característico do jogo.

Para fins do projeto, uma regra foi adaptada em relação à oficial, o ciclo de perguntas e respostas, que aqui é feito exclusivamente ao próximo jogador da ordem, enquanto na regra original, caso o primeiro jogador responda que não tem as cartas, a pergunta passa para os demais jogadores até que alguém possa responder. Essa alteração reflete uma variação observada em partidas informais do jogo e foi empregada por representar de forma fiel o comportamento dos jogadores no ambiente analisado.

Além disso, as partidas simuladas consideradas no desenvolvimento foram limitadas a quatro jogadores, definidos os jogadores `jogador_0` a `jogador_3` e permitindo que fatos e regras fossem estruturados de maneira consistente dentro desse formato.

O desenvolvimento seguiu um modelo incremental, no qual a base de conhecimento e o motor de inferência foram refinados em ciclos sucessivos. A cada iteração, regras eram revisadas e novas relações eram acrescentadas, garantindo que o sistema mantivesse coerência interna e integrasse corretamente a interface em Python ao núcleo lógico em Prolog. Esse processo estabeleceu os fundamentos necessários para as regras de decisão utilizadas pelo sistema. Dentre elas, destaca-se o conjunto de predicados responsáveis por avaliar cartas, determinar melhores combinações de perguntas e acusações.

## 4.3 Detalhamento das Regras do Sistema

Com a base de fatos definida e o modelo de decisão estruturado, torna-se essencial detalhar as regras implementadas no motor de inferência. Essas regras traduzem o raciocínio empregado no jogo em lógica formal, permitindo que o sistema realize deduções automáticas a partir das informações disponíveis. A seguir, serão apresentadas as principais regras desenvolvidas, destacando sua função, a lógica que representam e exemplos de aplicação em partidas simuladas.

A começar com as regras que auxiliam com a mecânica de anotações do jogo, a primeira regra analisada é registrar_pergunta/5, responsável pela criação de fatos a partir das perguntas realizadas pelos jogadores.

Figura 1 – Regra `registrar_pergunta/5`

```
registrar_pergunta(Jogador_a, Jogador_b, Carta_a, Carta_b,
   Resposta) :-
    tipos_diferentes(Carta_a, Carta_b),
    assertz(pergunta(Jogador_a, Jogador_b, Carta_a, Carta_b
       , Resposta)),
    (
        (Resposta == Carta_a ; Resposta == Carta_b) ->
        (
            assertz(tem_carta(Jogador_b, Resposta)),
            eliminar_carta(Resposta)
        ) ;
        (
            Resposta == true ->
            (
                assertz(pode_ter_carta(Jogador_b, Carta_a,
                   Carta_b))
            ) ;
            (
                Resposta == false ->
                (
                    assertz(nao_tem_carta(Jogador_b,
                       Carta_a)),
                    assertz(nao_tem_carta(Jogador_b,
                       Carta_b))
                )
            )
        )
    ).
```

**Fonte**: Autoria Própria (2025)

Essa regra tem a função de registrar informações sobre as perguntas e respostas realizadas durante o jogo. A partir dela, é possível anotar: os jogadores e as cartas envolvidos na pergunta; a resposta, que pode ser a carta no caso de uma pergunta direta ou um valor verdadeiro/falso em perguntas externas; se um jogador possui ou não determinadas cartas; e se ele pode ter uma das cartas questionadas em caso de perguntas externas com resposta visível limitada a verdadeiro ou falso. Diferentemente dos fatos originais, que representam elementos estáticos do jogo, esses fatos auxiliares permitem que o motor de inferência armazene dados dinâmicos do jogo e utilize essas informações para deduzir novas relações ou orientar sugestões de perguntas e acusações futuras. Outra regra capaz de realizar anotações de maneira direta sobre o jogo é `marcar_evidencia/1`, que recebe uma carta e cria um fato `carta_evidencia/2`, marcando a carta como evidência pessoal.

Além desses predicados, que realizam anotações de forma direta, existem também predicados como `inferir_posse_cartas/0` e `inferir_carta_crime/0`, responsáveis por percorrer os fatos auxiliares em busca de novas conclusões a partir das informações armazenadas.

Figura 2 – Regra `inferir_posse_cartas/0`

```
inferir_posse_cartas :-
    (
        tem_carta(_Jogador_a, Carta_a),
        pode_ter_carta(Jogador_b, Carta_a, Carta_b)
    ) ->
    (
        retract(pode_ter_carta(Jogador_b, Carta_a, Carta_b)),
        assertz(tem_carta(Jogador_b, Carta_B)),
        eliminar_carta(Carta_B)
    );
    (
        (
            tem_carta(_Jogador_a, Carta_B),
            pode_ter_carta(Jogador_B, Carta_a, Carta_B)
        ) ->
        (
            retract(pode_ter_carta(Jogador_B, Carta_a, Carta_B)),
            assertz(tem_carta(Jogador_B, Carta_a)),
            eliminar_carta(Carta_a)
        )
    ).
```

**Fonte**: Autoria Própria (2025)

A regra `inferir_posse_cartas/0` é um predicado que percorre os fatos dinâmicos do sistema para deduzir quais cartas os jogadores possuem, com base nas informações já registradas nas partidas. Ela analisa se uma `Carta_a` possuída por um `Jogador_a` aparece em uma resposta positiva de outro `Jogador_b` sobre possuir `Carta_a` ou `Carta_b`. Caso a condição seja satisfeita, o sistema remove a possibilidade de que `Jogador_b` possua `Carta_a`, afirma que ele possui `Carta_b` e elimina `Carta_b` do conjunto de cartas ainda não atribuídas. De maneira análoga, a regra verifica o cenário inverso, em que `Jogador_a` possui `Carta_b` e `Jogador_b` poderia ter `Carta_a` ou `Carta_b`, aplicando o mesmo processo de atualização. Dessa forma, o predicado permite que informações incompletas sejam eliminadas enquanto novas são descobertas.

A regra `inferir_carta_crime/0` percorre os fatos das cartas no sistema e identifica, para cada tipo (arma, lugar e suspeito), se há apenas uma carta restante que ainda não foi marcada como evidência, eliminada ou definida como carta do crime. Caso exista apenas uma carta disponível, ela é automaticamente marcada como carta do crime, permitindo que o motor de inferência utilize essa informação em deduções futuras, como nas regras `como_perguntar/1` e `como_acusar/1`, que utilizam dos status das cartas para ordenar sua importância para pergunta ou acusação usando um sistema de pesos. As regras `como_perguntar/1` e `melhor_pergunta/1` auxiliam no processo de perguntas, realizado ao longo da partida.

Figura 3 – Regra `como_perguntar/1`

```
como_perguntar(Resultado) :-
    findall(
        P-(Carta,Info),
        (
            carta(Carta,_),
            avaliar_carta_pergunta(Carta, P, Info)
        ),
        Lista),
    keysort(Lista, Ordenada),
    Resultado = Ordenada.
```

**Fonte**: Autoria Própria (2025)

O predicado `como_perguntar/1` é responsável por fornecer uma lista de cartas ordenadas por prioridade para perguntas durante o jogo. Ele percorre todas as cartas presentes na base de conhecimento e avalia cada uma delas por meio do predicado auxiliar `avaliar_carta_pergunta/3`, que retorna uma pontuação e uma descrição do estado da carta em relação ao jogador ou ao contexto do jogo. As condições a seguir avaliam o status da carta e

são executadas em ordem até que a carta se encaixe em um dos casos:

- Carta eliminada, de evidência ou do crime (peso 7): são cartas que já sabemos todas as informações e por isso são descartadas das perguntas. Atua como filtro impedindo que a carta caia em outra verificação.
- Não possuída pelos jogadores 2 e 3 (peso 1): A carta aparece em respostas negativas de perguntas feitas aos jogadores 2 e 3, só podendo pertencer ao jogador 1 ou ao crime, representando o melhor cenário para pergunta.
- Não possuída pelo jogador 2 ou pelo jogador 3 (peso 2): segundo melhor cenário para pergunta, aqui a carta aparece uma vez em uma resposta negativa.
- Não questionada (peso 3): carta que ainda não foi incluída em perguntas, podendo fornecer novas informações.
- Citada em pergunta ao jogador 1 (peso 4): carta mencionada em uma pergunta com resposta positiva onde ela não foi a resposta.
- Os jogadores 2 ou 3 podem ter (peso 5): carta que possivelmente está na mão de um jogador, mas ainda não confirmada. Aparece em uma pergunta externa, onde apenas sabemos que a resposta foi positiva.

Após a avaliação, o predicado `findall/3` coleta todos os pares pontuação-(carta, informação) em uma lista. Em seguida, `keysort/2` organiza a lista em ordem crescente de pontuação, de forma que as cartas mais estratégicas para serem perguntadas aparecem primeiro. O resultado final, atribuído a `Resultado`, é a lista ordenada de cartas com suas respectivas informações permitindo que a regra `melhor_pergunta/1` sugira de maneira eficiente as melhores cartas para a próxima pergunta.

Figura 4 – Regra `melhor_pergunta/1`

```
melhor_pergunta(Resultado) :-
    como_perguntar(Lista),
    Lista = [P_a-(Carta_a, Info_a) | Restante],

    member(P_b-(Carta_b, Info_b), Restante),
    tipos_diferentes(Carta_a, Carta_b),
    !,

    carta(Carta_a, Tipo_a),
    carta(Carta_b, Tipo_b),

    /* alternativas de Carta_a com mesmo peso P_a */
    findall(
        (Carta_alt_a, Info_alt_a),
        (
            member(P_a-(Carta_alt_a, Info_alt_a), Lista),
            Carta_alt_a \= Carta_a,
            carta(Carta_alt_a, Tipo_a)
        ),
        Outras_a),

    /* alternativas de Carta_b com mesmo peso P_b */
    findall(
        (Carta_alt_b, Info_alt_b),
        (
            member(P_b-(Carta_alt_b, Info_alt_b), Lista),
            Carta_alt_b \= Carta_b,
            carta(Carta_alt_b, Tipo_b)
        ),
        Outras_b),

    Resultado = [(Carta_a, Info_a, Outras_a),  (Carta_b,
       Info_b, Outras_b)].
```

**Fonte**: Autoria Própria (2025)

O predicado `melhor_pergunta/1` é responsável por selecionar o melhor par de cartas para formular perguntas, utilizando como base a lista ordenada gerada por `como_perguntar/1`. Seu funcionamento pode ser descrito em etapas:

- Primeiramente, `melhor_pergunta/1` chama `como_perguntar(Lista)`, obtendo uma lista de todas as cartas avaliadas, cada uma associada a um peso (`P`) e informações sobre seu status (`Info`). Essa lista já está ordenada em ordem crescente de peso, priorizando as cartas mais relevantes para dedução.

- Em seguida, a cabeça da lista (`P_a-(Carta_a, Info_a)`) é selecionada como a primeira carta candidata, enquanto `member(P_b-(Carta_b, Info_b), Restante)` busca uma segunda carta de tipo diferente (`tipos_diferentes(Carta_a, Carta_b)`).
- Para cada uma dessas cartas selecionadas, o predicado utiliza `findall/3` para identificar outras alternativas de mesmo peso e tipo (`Outras_a` e `Outras_b`), permitindo que o usuário possa escolher entre cartas equivalentes.
- Por fim, `Resultado` reúne as duas cartas principais e as suas alternativas em uma lista estruturada, que servirá como referência para que o usuário formule sua pergunta.

O predicado `como_responder/3` é responsável por determinar qual carta deve ser apresentada como resposta a uma pergunta. Ele recebe duas cartas envolvidas na pergunta (`Carta_a` e `Carta_b`) e retorna uma indicação de qual delas deve ser mostrada, ou se não é possível ajudar.

O funcionamento é baseado nas informações já registradas pelo sistema: Não está nas evidências: Se nenhuma das cartas foi marcada como evidência, o predicado retorna que não é possível ajudar; Possui uma das cartas: Se apenas uma das cartas perguntadas consta em suas evidências, essa será a escolhida para a resposta; Caso ambas as cartas sejam evidências, o sistema utiliza um critério de prioridade baseado no tipo da carta (`suspeito > lugar > arma`) para decidir qual mostrar. Essa escolha leva em consideração que cartas de tipo com maior quantidade fornecem menos informação nova ao jogador adversário.

Além das regras detalhadas neste capítulo, o sistema conta com diversos outros predicados auxiliares que dão suporte ao funcionamento do motor de inferência, realizando tarefas como verificação e criação de fatos, consultas de anotações e limpeza do estado da base de conhecimento. Essas regras complementares garantem que o SE opere de forma coesa e eficiente, mas não são apresentadas detalhadamente neste texto por possuírem papel auxiliar ao funcionamento do SE. O código completo, contendo todos os predicados do projeto, pode ser consultado no repositório indicado nos Apêndices.

Finalizando as regras que auxiliam nas quatro mecânicas do jogo Clue Suspeitos (Anotar, Perguntar, Responder e Acusar), os predicados `como_acusar/1` e `melhor_acusacao/1` funcionam de maneira semelhante à `como_perguntar/1` e `melhor_pergunta/1`, auxiliando o jogador na escolha das cartas que formam a acusação do crime (uma arma, um lugar e um suspeito). A diferença está nos critérios de ordenação das cartas, onde em `como_acusar/1`, as cartas são priorizadas de forma a minimizar a chance de pertencerem a algum jogador, enquanto em `como_perguntar/1` buscavam-se cartas com maior probabilidade de estar com o `jogador_1`.

Por sua vez, `melhor_acusacao/1` opera de maneira semelhante a `melhor_pergunta/1`, retornando as três cartas do crime organizadas por nível de informação e considerando cartas equivalentes.

## 4.4 Arquitetura e Tecnologias do Sistema

Com a base conceitual e a modelagem do conhecimento definidas, passou-se ao projeto da arquitetura do sistema. O sistema especialista foi estruturado em três camadas principais, garantindo organização modular e clareza na comunicação entre componentes. A primeira camada é a interface em Python (`interface.py`), responsável pelo controle da interação com o usuário via linha de comando (CLI), coleta das informações de jogo e exibição dos resultados das deduções. A segunda camada, o módulo de controle `controlador.py`, atua como mediador entre a interface e o Prolog, gerenciando a comunicação e encaminhando consultas ao motor de inferência. Por fim, a terceira camada é o motor de inferência (`motor_inferencia.pl`), desenvolvido em Prolog, que compre dupla função: representa a base de conhecimento do jogo por meio de fatos e regras e, ao mesmo tempo, processa essas informações, atuando como motor de inferência, realizando deduções automaticamente e eliminando possibilidades incoerentes conforme as consultas enviadas pelo módulo de controle. A comunicação entre Python e Prolog é realizada por meio da biblioteca `(PySWIP)`, permitindo a manipulação do Prolog em tempo real. Essa arquitetura modular facilita a manutenção, a extensibilidade do sistema e garante que a lógica de inferência permaneça separada da interface com o usuário.

Para a implementação do projeto foram selecionadas as seguintes tecnologias:

- **Prolog:** Linguagem lógica utilizada para o desenvolvimento do Motor de Inferência. Seu paradigma declarativo permite representar o conhecimento por meio de fatos e regras, executando buscas encadeadas que refletem diretamente o processo de dedução utilizado em Sistemas Especialistas.
- **SWI-Prolog:** Implementação do Prolog escolhida para este trabalho. Foi selecionada devido à sua facilidade de instalação, ampla documentação, suporte a módulos e compatibilidade com integração externa, característica essencial para a comunicação com Python.
- **Python:** Utilizado na implementação da interface de interação com o usuário, atuando como mediador entre o jogador e o motor de inferência. A versão escolhida foi o `Python 3.14.0`, e a integração com SWI-Prolog foi realizada por meio da biblioteca `PySwip`

`9.2.9`. Essa integração possibilitou enviar consultas Prolog, atualizar dinamicamente a base de conhecimentos e exibir os resultados inferidos de maneira direta para o usuário.

## 4.5 Validação e Testes

A etapa de Validação e Testes teve como objetivo verificar se o SE se comporta de maneira consistente, correta e alinhada às regras do jogo Clue Suspeitos. Para isso, foram elaborados cenários simulados que reproduzem situações reais observadas em partidas, com diversas combinações de cartas, hipóteses, informações parciais e sequências de jogadas. A partir desses cenários, o sistema foi submetido a consultas controladas, avaliando se as deduções lógicas produzidas pelo motor de inferência correspondiam ao comportamento esperado. Esse processo permitiu analisar a capacidade do sistema de eliminar possibilidades incoerentes e convergir para conclusões válidas.

Para verificar a consistência e o correto funcionamento das regras implementadas em Prolog, utilizou-se o framework de testes unitários plunit, disponibilizado nativamente pelo SWI-Prolog. Esse mecanismo possibilitou a criação de módulos de teste independentes, nos quais cada regra (ou conjunto de regras) foi avaliada individualmente. Os casos de teste foram estruturados de forma a representar situações típicas do jogo, como deduções lógicas baseadas nas distribuições de cartas e de informações expostas por perguntas. A abordagem sistemática proporcionada pelo plunit permitiu identificar inconsistências na base de conhecimento, validar o comportamento incremental das regras e assegurar que o motor de inferência mantivesse coerência ao longo das iterações do desenvolvimento.

## 4.6 Escopo e Limitações da Metodologia

A metodologia adotada neste trabalho define um conjunto específico de condições e simplificações que orientam o funcionamento do sistema especialista. O modelo considera exclusivamente partidas com quatro jogadores fixos, representados de forma estática na base de conhecimento, o que facilita a estruturação das regras e a interpretação dos fatos gerados durante o jogo. Além disso, a dinâmica original do Clue Suspeitos foi adaptada para um fluxo simplificado de perguntas: sempre que um jogador realiza uma pergunta, apenas o próximo jogador na ordem deve respondê-la.

O sistema também não modela o comportamento estratégico dos jogadores adversários.

Assim, não são considerados blefes, respostas intencionais para induzir erro, nem qualquer forma de manipulação baseada na interpretação das informações que outros jogadores possuam sobre o estado da partida. Todas as inferências são conduzidas de maneira estritamente lógica, derivadas exclusivamente dos fatos registrados e das regras implementadas, sem o uso de mecanismos probabilísticos ou heurísticos.

Essas restrições estabelecem um escopo bem delimitado, que permite analisar com clareza o comportamento do motor de inferência e demonstrar a aplicabilidade de um sistema especialista ao domínio do jogo.

# 5 RESULTADOS

Este capítulo apresenta os resultados do desenvolvimento do Sistema Especialista para o jogo Clue Suspeitos. As análises incluem a execução das mecânicas do jogo pelo sistema, o funcionamento do motor de inferência em diferentes cenários simulados, a evolução do estado das cartas e o desempenho nas sugestões de perguntas e acusações. Além disso, são discutidas limitações observadas e aspectos de validação do protótipo.

## 5.1 Configuração dos Cenários de Teste

Para validar o funcionamento do sistema, foram definidos cenários de teste simulando partidas com diferentes distribuições de cartas e sequências de jogadas. Cada cenário considerou quatro jogadores fixos (`jogador_0` a `jogador_3`), permitindo observar o comportamento do motor de inferência diante de informações parciais. A Tabela 2 apresenta um exemplo de distribuição de cartas utilizada, incluindo as cartas do crime e a posse inicial de cada jogador.

Tabela 2 – Distribuição de cartas no cenário de teste

| **Jogador** | **Carta 1** | **Carta 2** | **Carta 3** |
|---|---|---|---|
| Crime | Scarlet (suspeito) | Spa (local) | Corda (arma) |
| jogador_0 | Castiçal (suspeito) | Cozinha (local) | Hall (local) |
| jogador_1 | Faca (arma) | Cozinha (local) | Mustard (suspeito) |
| jogador_2 | Revólver (arma) | Sala de Estar (local) | Peacock (suspeito) |
| jogador_3 | Green (suspeito) | Sala de Jantar (local) | White (suspeito) |

**Fonte**: Autoria Própria (2025)

O ambiente de teste incluiu Python 3.14.0 para interface e controle, SWI-Prolog 9.2.9 para o motor de inferência e integração via `PySWIP`. O hardware utilizado foi um processador AMD Ryzen 5 5600U, 20 GB de RAM, rodando Windows 10 x64. A base de conhecimento continha todas as 15 cartas do jogo (quatro armas, cinco locais e seis suspeitos), estruturadas em fatos e regras em Prolog. Os cenários incluíram perguntas diretas e externas, respostas afirmativas e negativas, permitindo avaliar a capacidade do motor de inferência em atualizar o estado do jogo e fornecer sugestões consistentes.

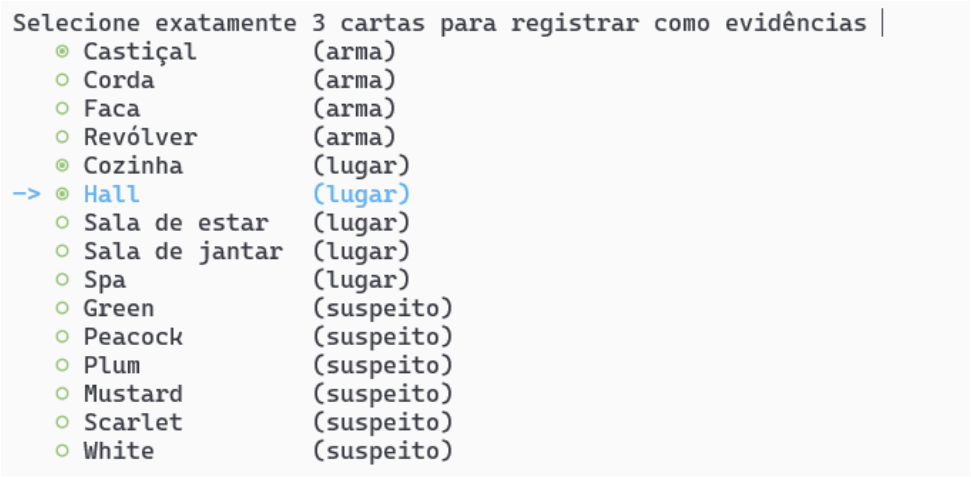
O código-fonte completo do SE desenvolvido neste trabalho está disponível em um repositório público, acessível em: https://github.com/queirozPedro/clue-se. O repositório reúne os módulos de interface, controle e motor de inferência, além de instruções para instalação e

execução, permitindo a inspeção do código e a reprodução dos cenários de teste apresentados neste capítulo.

## 5.2 Execução das Mecânicas do Sistema

Para ilustrar o funcionamento do sistema, foram utilizados cenários de teste como os representados na Tabela 2. Inicialmente, o sistema executa `reiniciar_estado/0`, eliminando predicados dinâmicos existentes e garantindo integridade na execução. Em seguida, solicita-se ao usuário a seleção de cartas de evidências, registradas pelo predicado `marcar_evidencia/1`, (Figura 5).

Figura 5 – Print do terminal mostrando o preenchimento das evidências




**Fonte**: Autoria Própria (2025)

Após a seleção, o jogador acessa o menu principal do sistema, exibindo as opções de visualização de anotações, registro de perguntas, sugestões de resposta e sugestões de acusação (Figura 6). A interface Python converte as ações do usuário em consultas para o motor de inferência em Prolog, atualizando a base de conhecimento em tempo real.

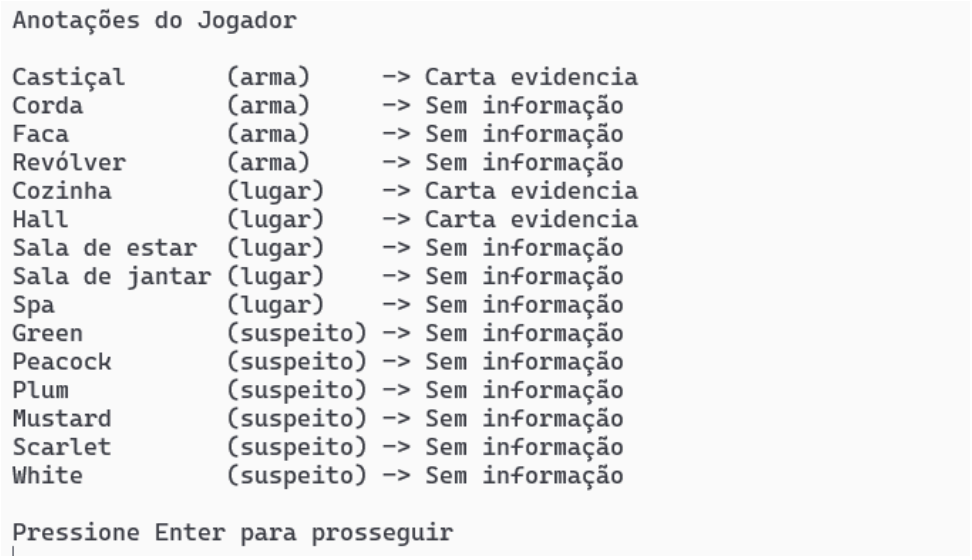
Figura 6 – Print do terminal mostrando o menu do sistema


**Fonte**: Autoria Própria (2025)

Ao selecionar "Visualizar anotações", o usuário observa o estado inicial das cartas,

evidências e possibilidades abertas (Figura 7). Esta visualização permite acompanhar a evolução da partida de forma organizada.

Figura 7 – Print do terminal mostrando as anotações ao início do jogo




**Fonte**: Autoria Própria (2025)

A opção "Registrar pergunta"apresenta ao usuário sugestões estratégicas para formular perguntas. Na parte superior, a regra `melhor_pergunta` indica combinações com maior potencial informativo; na parte inferior, a regra `como_perguntar` organiza as cartas em ordem de prioridade (Figura 8). Essa interface orienta o usuário e registra automaticamente a pergunta no histórico.

Figura 8 – Print do terminal mostrando a interface de pergunta com baixo nível de informação

```
? Selecione os envolvidos na pergunta: Jogador 0 para Jogador 1

Cartas recomendadas para a pergunta:
(Cartas agrupadas com outras do mesmo tipo possuem a mesma chanc
e de estarem com o outro jogador)

Tipo: Arma
Corda          -> Não questionada
Faca           -> Não questionada
Revólver       -> Não questionada

Tipo: Lugar
Sala de estar  -> Não questionada
Sala de jantar -> Não questionada
Spa            -> Não questionada

Registre sua pergunta
-> ○ Corda          (arma)     -> Não questionada
   ○ Faca           (arma)     -> Não questionada
   ○ Revólver       (arma)     -> Não questionada
   ○ Sala de estar  (lugar)    -> Não questionada
   ○ Sala de jantar (lugar)    -> Não questionada
   ○ Spa            (lugar)    -> Não questionada
   ○ Green          (suspeito) -> Não questionada
   ○ Peacock        (suspeito) -> Não questionada
   ○ Plum           (suspeito) -> Não questionada
   ○ Mustard        (suspeito) -> Não questionada
   ○ Scarlet        (suspeito) -> Não questionada
   ○ White          (suspeito) -> Não questionada
   ○ Castiçal       (arma)     -> Carta de evidencia
   ○ Cozinha        (lugar)    -> Carta de evidencia
   ○ Hall           (lugar)    -> Carta de evidencia
```

**Fonte**: Autoria Própria (2025)

Na sequência, a pergunta sugerida pelo sistema (Corda e Sala de Estar) foi realizada, e o jogador 1 respondeu "Não posso ajudar"(Figura 9). O motor de inferência registrou automaticamente essa resposta, eliminando essas cartas da posse do jogador consultado e atualizando a prioridade de dedução para futuras perguntas.

Figura 9 – Print do terminal mostrando a resposta do jogador 1 à pergunta sugerida pelo sistema.

```
Registre a resposta da pergunta
   Corda
   Sala de estar
-> Não posso ajudar
```

**Fonte**: Autoria Própria (2025)

Perguntas subsequentes realizadas pelo jogador 1 e pelo jogador 2 modificaram o status das cartas, com respostas afirmativas e negativas. O motor de inferência atualizou as prioridades de dedução, eliminou cartas confirmadas e priorizou aquelas com maior probabilidade de estarem com outro jogador ou ser carta do crime, como Spa e Scarlet. A evolução do estado das cartas após essas interações é apresentada na Figura 10.

Figura 10 – Print do terminal mostrando a evolução dos status após perguntas.

```
? Selecione os envolvidos na pergunta: Jogador 0 para Jogador 1

Cartas recomendadas para a pergunta:
(Cartas agrupadas com outras do mesmo tipo possuem a mesma chanc
e de estarem com o outro jogador)

Tipo: Lugar
Spa            -> Não possuida pelo jogador 3

Tipo: Suspeito
Scarlet        -> Não possuida pelo jogador 3

Registre sua pergunta
-> ○ Spa            (lugar)    -> Não possuida pelo jogador 3
   ○ Scarlet        (suspeito) -> Não possuida pelo jogador 3
   ○ Faca           (arma)     -> Não questionada
   ○ Sala de jantar (lugar)    -> Não questionada
   ○ Green          (suspeito) -> Não questionada
   ○ Peacock        (suspeito) -> Não questionada
   ○ Plum           (suspeito) -> Não questionada
   ○ Mustard        (suspeito) -> Não questionada
   ○ White          (suspeito) -> Não questionada
   ○ Corda          (arma)     -> Não possuida pelo jogador 1
   ○ Sala de estar  (lugar)    -> Não possuida pelo jogador 1
   ○ Revólver       (arma)     -> Carta eliminada
   ○ Castiçal       (arma)     -> Carta de evidencia
   ○ Cozinha        (lugar)    -> Carta de evidencia
   ○ Hall           (lugar)    -> Carta de evidencia
```

**Fonte**: Autoria Própria (2025)

Em outro momento, o jogador 0 foi questionado pelo jogador 1 sobre Hall e Castiçal. A resposta do jogador 0, registrada automaticamente pelo sistema, atualizou o histórico de interações e ajustou as possibilidades de posse das cartas, eliminando combinações inconsistentes e refinando as sugestões de perguntas e acusações futuras (Figura 11).

Figura 11 – Print do terminal mostrando a resposta do `jogador_0` à pergunta sobre Hall e Castiçal.

```
? Selecione uma opção: Sugestão de resposta
? Selecione as cartas envolvidas na pergunta ['Castiçal
(arma)', 'Hall            (lugar)']
Resposta: Hall
```

**Fonte**: Autoria Própria (2025)

Por fim, o sistema foi testado na funcionalidade de acusações. Com base nas informações inferidas durante a partida, o motor de inferência sugeriu combinações prováveis de cartas do crime, permitindo ao usuário realizar acusações fundamentadas (Figura 12).

Figura 12 – Print do terminal mostrando as sugestões de acusação geradas pelo sistema.

```
Palpite de acusação baseado nas informações coletadas.
Cartas agrupadas com outras do mesmo tipo representam a mesma ch
ance de estarem envolvidas no crime.

Carta do tipo arma:
Corda          -> Outros 2 jogadores não tem essa carta

Carta do tipo lugar:
Sala de estar  -> Outros 2 jogadores não tem essa carta
Spa            -> Outros 2 jogadores não tem essa carta

Carta do tipo suspeito:
Scarlet        -> Outros 2 jogadores não tem essa carta
```

**Fonte**: Autoria Própria (2025)

Após a seleção das evidências, o sistema encontra-se pronto para conduzir a partida, registrando perguntas, respostas e atualizando automaticamente o estado do jogo a cada interação. Esse processo demonstra a capacidade do motor de inferência de integrar informações parciais, ajustar deduções dinamicamente e fornecer ao usuário orientações consistentes para perguntas e acusações. Dessa forma, os cenários de teste cumprem seu papel de validar o comportamento do sistema em condições controladas, evidenciando sua robustez e a efetividade da lógica implementada.

## 5.3 Avaliação Geral do Desempenho

Os testes com cenários simulados demonstraram que o motor de inferência funciona de maneira consistente com a lógica do jogo Clue Suspeitos®. O sistema consegue identificar corretamente as cartas do crime, eliminar combinações impossíveis e atualizar o estado do jogo à medida que novas informações são inseridas. A integração entre Python e Prolog, mediada pelo PySwip, mostrou-se estável e eficiente, permitindo que perguntas, respostas e acusações sejam registradas e processadas durante a execução da partida.

Durante o processo de desenvolvimento do trabalho, algumas limitações técnicas dos SEs baseados em regras tornaram-se evidentes. Uma das principais dificuldades foi a falta de um especialista consolidado na área de aplicação escolhida, o que exigiu que o próprio desenvolvedor se aprofundasse no tema e assumisse o papel de especialista para realizar o processo de extração, estruturação e validação do conhecimento, como bem descreve (Negnevitsky, 2005). Essa limitação revela uma grande fragilidade desse tipo de sistema: sua dependência de especialistas humanos para alimentar sua base de conhecimento. Como alertam Costa e Silva (2005), “Por

não ser um produto concreto o conhecimento pode simplesmente sumir ou não estar presente quando for necessitado".

Além disso, o sistema funciona exclusivamente de maneira lógica, realizando deduções sobre as informações disponíveis sem interpretar estratégias ou intenções dos adversários, como blefes ou padrões de comportamento, aspectos típicos da dinâmica de jogadores humanos. Apesar dessa limitação, ele cumpre seu objetivo principal de apoiar deduções racionais durante a partida, fornecendo orientações coerentes para perguntas e acusações.

Em síntese, os resultados obtidos evidenciam que a metodologia adotada possibilitou a construção de um Sistema Especialista funcional, capaz de simular deduções automatizadas de forma confiável. Essa base sólida permite futuras extensões, incluindo a incorporação de estratégias avançadas e comportamento mais próximo de agentes humanos, ampliando a aplicabilidade do sistema em cenários de maior complexidade.

# 6 CONCLUSÕES E TRABALHOS FUTUROS

O desenvolvimento deste trabalho envolveu inicialmente a compreensão do funcionamento dos Sistemas Especialistas e de seu papel na modelagem de jogos, destacando como a Inteligência Artificial pode ser aplicada para resolver problemas estratégicos, gerar comportamentos de inimigos e controlar NPCs. Parte fundamental do processo foi a análise do jogo Clue Suspeitos®, para identificar como as regras e interações entre jogadores poderiam ser traduzidas em conhecimento lógico.

A criação das regras constituiu o núcleo do Sistema Especialista, abrangendo as quatro operações fundamentais: anotações, formulação de perguntas, respostas e acusações. Cada regra foi projetada para orientar como as decisões devem ser tomadas do ponto de vista de um especialista, refletindo a lógica e a estratégia subjacentes a cada ação. Para tanto, foi necessário que eu, enquanto desenvolvedor e especialista do projeto, compreendesse profundamente o funcionamento do jogo, aprendendo suas nuances e estratégias. A partir desse conhecimento, as regras foram formalizadas em Prolog, garantindo que o motor de inferência pudesse atualizar corretamente o estado do jogo, priorizar deduções relevantes e fornecer orientações consistentes, simulando o raciocínio de um jogador experiente.

O Prolog teve um papel fundamental no desenvolvimento do projeto, oferecendo uma base sólida para a implementação do Sistema Especialista. Sua natureza declarativa permitiu a formalização das regras de decisão de forma clara e estruturada, facilitando a modelagem do raciocínio dedutivo necessário para as quatro operações centrais do sistema. Além disso, o uso de Prolog possibilitou que o motor de inferência atualizasse dinamicamente o estado do jogo e realizasse deduções consistentes, refletindo com precisão o comportamento de um especialista humano.

Os testes realizados demonstraram que o sistema realiza inferências coerentes e mantém desempenho estável mesmo em diferentes cenários de teste, confirmando a efetividade do motor de inferência implementado. Dessa forma, o trabalho atingiu seus objetivos, desenvolvendo um Sistema Especialista com motor em Prolog, elaborando uma base de conhecimento a partir da modelagem das regras, criando um sistema capaz de interagir com o usuário e validando seu desempenho por meio de diferentes cenários de teste. Foi possível comparar os resultados por meio de simulações observáveis das decisões lógicas efetuadas, apresentar as explicações dedutivas geradas pelo motor de inferência e documentar todo o processo de desenvolvimento, destacando os desafios enfrentados e os aprendizados obtidos.

Além disso, o desenvolvimento deste Sistema Especialista se destaca por possuir poucos trabalhos semelhantes na literatura, evidenciando sua originalidade e a contribuição para a área de sistemas especialistas aplicados a jogos. Essa característica reforça a relevância do estudo e o potencial de uso como referência para futuras pesquisas ou aplicações em contextos semelhantes. A estrutura desenvolvida também oferece aplicabilidade prática em contextos educativos, de pesquisa em Inteligência Artificial e como base para desenvolvimento de sistemas estratégicos para jogos com agentes inteligentes.

Como trabalhos futuros, sugere-se o desenvolvimento de um sistema que leve em conta não apenas as informações recebidas, mas também as próprias informações do agente. Esse aprimoramento permitiria ao sistema formular perguntas e respostas de modo estratégico, entregando o mínimo de informação aos adversários enquanto maximiza o ganho de conhecimento, aproximando o comportamento do sistema ao de um agente competitivo e estratégico. Além disso, seria interessante criar agentes capazes de representar NPCs inimigos, permitindo partidas simuladas contra o usuário e proporcionando um ambiente de teste mais completo e desafiador.

Por fim, mesmo com as limitações apontadas, o sistema desenvolvido forneceu uma base sólida para a dedução das ações do jogo, permitindo a realização de inferências consistentes e a aplicação prática das regras modeladas. Essa estrutura serve como ponto de partida para futuras expansões, possibilitando o desenvolvimento de agentes mais estratégicos e de cenários simulados mais complexos.

# REFERÊNCIAS


BARRETO, L.; PREZOTO, M. **Introdução a sistemas especialistas**. Universidade Estadual de Campinas–UNICAMP–Dissertação de Mestrado em Tecnologia para Sistemas e Fenômenos Complexos. São Paulo, 2010.

BRATKO, I. **Prolog programming for artificial intelligence**. [S.l.]: Pearson education, 2001.

BUCHANAN, B.; SUTHERLAND, G.; FEIGENBAUM, E. A. Heuristic dendral: A program for generating explanatory hypotheses. **Organic Chemistry**, v. 30, 1969.

BUCHANAN, B. G.; SMITH, R. G. Fundamentals of expert systems. **Annual review of computer science**, v. 3, n. 1, p. 23–58, 1988.

CLOCKSIN, W. F.; MELLISH, C. S. **Programming in PROLOG**. [S.l.]: Springer Science & Business Media, 2003.

COLMERAUER, A.; ROUSSEL, P. The birth of prolog. In: **History of programming languages—II**. [S.l.: s.n.], 1996. p. 331–367.

COSTA, W. S.; SILVA, S. C. M. Aquisição de conhecimento: O grande desafio na concepção de sistemas especialistas. **Holos**, Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, v. 2, p. 37–46, 2005.

GOMES, D. d. S. Inteligência artificial: conceitos e aplicações. **Revista Olhar Científico**, v. 1, n. 2, p. 234–246, 2010.

HASBRO. **Clue Card Game Instructions**. 2018. https://instructions.hasbro.com/en-au/instruction/clue-card-game. Acesso em: 15 maio 2025.

HSU, F.-h. Ibm's deep blue chess grandmaster chips. **IEEE micro**, IEEE, v. 19, n. 2, p. 70–81, 1999.

JONES, C. R.; BERGEN, B. K. **Large Language Models Pass the Turing Test**. 2025. Disponível em: https://arxiv.org/abs/2503.23674.

KOEHLER, C. **Uma abordagem probabilistica para sistemas especialistas**. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Florianópolis, 1998.

LAPEYRADE, S.; REY, C. Non-player character decision-making with prolog and ontologies. In: IEEE. **2023 IEEE Conference on Games (CoG)**. [S.l.], 2023. p. 1–2.

MCCARTHY, J. *et al.* A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. **AI magazine**, v. 27, n. 4, p. 12–12, 2006.

MENDES, R. D. Inteligência artificial: sistemas especialistas no gerenciamento da informação. **Ciência da Informação**, SciELO Brasil, v. 26, p. 39–45, 1997.

MERRITT, D. **Building expert systems in Prolog**. [S.l.]: Springer Science & Business Media, 2012.

NEGNEVITSKY, M. **Artificial intelligence: a guide to intelligent systems**. [S.l.]: Pearson education, 2005.

OLIVEIRA, G. S.; SILVA, A. F. da. Prolog: A linguagem, a máquina abstrata de warren e implementações. **Revista de Informática Teórica e Aplicada**, v. 20, n. 2, p. 214–246, 2013.

RIBEIRO, B. *et al.* Inteligência artificial em jogos digitais. **UNICAMP, sd Disponível em: http://www. dca. fee. unicamp. br/˜ martino/disciplinas/ia369/trabalhos/t4g3. Consultado em**, v. 25, 2019.

RUSSELL, P. N. *et al.* Artificial intelligence: a modern approach by stuart. **Russell and Peter Norvig contributing writers, Ernest Davis...[et al.]**, 2010.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. **IBM Journal of research and development**, IBM, v. 3, n. 3, p. 210–229, 1959.

SELLMER, D. *et al.* Sistema especialista para apoiar a decisão na terapia tópica de úlceras venosas. **Revista Gaúcha de enfermagem**, SciELO Brasil, v. 34, p. 154–162, 2013.

SHORTLIFFE, E. **Computer-based medical consultations: MYCIN**. [S.l.]: Elsevier, 2012. v. 2.

SILVER, D. *et al.* Mastering the game of go with deep neural networks and tree search. **nature**, Nature Publishing Group, v. 529, n. 7587, p. 484–489, 2016.

SIQUEIRA, J.; ROCIO, V. Sistema pericial em prolog no diagnóstico do potencial agrícola de solos. In: **Atas do VI Congresso Ibérico de Agro-Engenharia, Évora, Portugal**. [S.l.: s.n.], 2011. v. 5.

WIELEMAKER, J. An overview of the swi-prolog programming environment. **WLPE**, v. 3, p. 1–16, 2003.