UNIVERSIdade FEEVALE

FELIPE KURKOWSKI WEBER

AGENTE RACIONAL PARA TEXAS HOLD’EM pôquer

Novo Hamburgo

2010

felipe kurkowski weber

agente racional para texas hold’em pôquer

Trabalho de Conclusão de Curso

apresentado como requisito parcial

à obtenção do grau de Bacharel em

Ciência da Computação pela

Universidade Feevale

Orientador: Ricardo Ferreira de Oliveira

Novo Hamburgo

2010

Resumo

Por muito tempo, os jogos com informações completas foram única e amplamente utilizados para a pesquisa de técnicas de inteligência artificial. Jogos como damas e gamão já foram praticamente solucionados e até mesmo o xadrez, que possui complexidade elevada, se encontra em fase muito avançada de pesquisa. Em contraposição, o pôquer começou a ser estudado muito recentemente e apresenta um domínio de estudo onde uma quantidade considerável das informações necessárias para a tomada de decisões não é visível ao jogador. Adicionalmente, a existência de elementos estocásticos causa variância nos resultados e, consequentemente, na análise das jogadas. Essas dificuldades, apesar de desafiadoras, impulsionam o estudo da inteligência artificial para os jogos não determinísticos e sem informações completas, que representam mais claramente alguns problemas encontrados na vida real. Além da insuficiência de informação e da aleatoriedade, o jogo engloba outros fatores pertinentes ao estudo da inteligência artificial, como a existência de múltiplos agentes, a capacidade do jogador em lidar com a indução de erros e a necessidade de modelar seus oponentes de acordo com suas particularidades. Sendo assim, este trabalho tem como objetivo a criação de um agente racional para *Texas Hold’em* pôquer capaz de justificar suas decisões combinando dois elementos: a análise matemática e a modelagem de oponentes. Para tanto, aplicar-se-á técnicas de inteligência artificial relacionadas a agentes inteligentes. Por fim, avaliar-se-á a capacidade do protótipo de agente proposto de jogar em nível avançado, compreendendo as diversas situações presentes no jogo e explorando seus adversários de maneira adequada.

Palavras-chave: Inteligência Artificial. Agentes Racionais. Modelagem de Oponentes. Texas Hold’em Pôquer.

ABSTRACT

For a long time, perfect information games were uniquely and widely used for artificial intelligence research. Games like checkers and backgammon already have been practically solved and even chess, which has high complexity, is at an advanced research stage. In opposition, poker has started to be studied only recently and offers a domain where a considerable amount of the information needed for decision-making is not available to the player. Additionally, the presence of stochastic elements adds variance to the results and, consequently, to play analysis. These obstacles, although challenging, encourage the study of imperfect information non-deterministic games which represent more clearly some of the problems found in the real world. In addition to imperfect information and randomness, the game provides other pertinent factors in the study of artificial intelligence, including multiple competing agents, error induction and opponent modeling. Therefore, this work aims to develop a rational agent for Texas Hold'em capable of justifying its decisions by combining two elements: mathematical analysis and opponent modeling. In order to do this, artificial intelligence techniques related to intelligent agents will be used. Ultimately, the proposed agent's capability of high level play will be evaluated, measuring if the system is able to understand the many distinct situations present in the game and exploit its opponent efficiently.

Keywords: Artificial Intelligence. Rational Agents. Opponent Modeling. Texas Hold’em Poker.

LISTA DE QUADROS

[Quadro 1 Características do pôquer 35](#_Toc280930923)

[Quadro 2 Análise DIVAT para P1 38](#_Toc280930924)

LISTA DE tabelas

[Tabela 1 Índices de força 56](#_Toc280930927)

[Tabela 2 Resultados do ReduxBot em situação genérica 83](#_Toc280930928)

[Tabela 3 Resultados do EssentialBot em situação genérica 88](#_Toc280930929)

[Tabela 4 Ganhos do EssentialBot em cada confronto individual 90](#_Toc280930930)

LISTA DE FIGURAS

[Figura 1 Arquitetura de um agente 15](#_Toc280930931)

[Figura 2 Arquitetura reativa 18](#_Toc280930932)

[Figura 3 Arquitetura deliberativa 20](#_Toc280930933)

[Figura 4 Arquitetura híbrida 21](#_Toc280930934)

[Figura 5 Fórmula de EV 27](#_Toc280930935)

[Figura 6 EV do jogo de dados 28](#_Toc280930936)

[Figura 7 Lucratividade do *call* 28](#_Toc280930937)

[Figura 8 Valor mínimo para pagar lucrativamente 30](#_Toc280930938)

[Figura 9 Posições em uma mesa com seis jogadores 32](#_Toc280930939)

[Figura 10 Arquitetura do Loki 41](#_Toc280930940)

[Figura 11 Fórmulas de avaliação de mãos 42](#_Toc280930941)

[Figura 12 Arquitetura do Poki 44](#_Toc280930942)

[Figura 13 Composição do PsOpti2 47](#_Toc280930943)

[Figura 14 Árvore de contexto 50](#_Toc280930944)

[Figura 15 Arquitetura do ReduxBot 57](#_Toc280930945)

[Figura 16 Diagrama de classes abstraído 61](#_Toc280930946)

[Figura 17 Estimativa de *range* para aumentos 64](#_Toc280930947)

[Figura 18 Definindo o *range* de um oponente que ainda não agiu 65](#_Toc280930948)

[Figura 19 Procedimento melhor aposta 67](#_Toc280930949)

[Figura 20 Árvore simplificada 70](#_Toc280930950)

[Figura 21 Árvore com histogramas 71](#_Toc280930951)

LISTA DE gráficos

[Gráfico 1 ReduxBot em situação genérica 82](#_Toc280930952)

[Gráfico 2 Enfrentamento Reduxbot versus ORenegado 84](#_Toc280930953)

[Gráfico 3 ReduxBot contra um oponente explorador 85](#_Toc280930954)

[Gráfico 4 ReduxBot versus AlwaysCallBot 86](#_Toc280930955)

[Gráfico 5 EssentialBot em situação genérica 87](#_Toc280930956)

[Gráfico 6 EssentialBot versus ReduxBot 89](#_Toc280930957)

[Gráfico 7 EssentialBot *versus* AlwaysCallBot 91](#_Toc280930958)

[Gráfico 8 Histograma de frequência 92](#_Toc280930959)

LISTA DE ABREVIATURAS E SIGLAS

|  |  |
| --- | --- |
| API | Application Programming Interface |
| BTN | Button |
| BB | Big Blind |
| CC | Cold Call |
| CO | Cutoff |
| CPRG | Computer Poker Research Group |
| IA | Inteligência Artificial |
| MO | Modelagem de Oponentes |
| MP | Middle Position |
| PFR | Pre-flop Raise |
| RAM | Random Access Memory |
| SB | Small Blind |
| UTG | Under The Gun |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |
|  |

SUMÁRIO

[1 agentes 15](#_Toc280930964)

[1.1 Características de um agente 16](#_Toc280930965)

[1.2 Percepção 16](#_Toc280930966)

[1.3 Arquiteturas 17](#_Toc280930967)

[1.3.1 Arquiteturas reativas 17](#_Toc280930968)

[1.3.2 Arquiteturas deliberativas 19](#_Toc280930969)

[1.3.3 Arquiteturas híbridas 20](#_Toc280930970)

[1.4 Agentes racionais 22](#_Toc280930971)

[1.4.1 Avaliando a melhor decisão 23](#_Toc280930972)

[1.5 Aplicações de agentes 23](#_Toc280930973)

[1.5.1 Controle de tráfico aéreo 23](#_Toc280930974)

[1.5.2 Filtragem de informação 24](#_Toc280930975)

[1.5.3 Comércio Eletrônico 24](#_Toc280930976)

[2 TEXAS HOLD’EM PÔQUER 26](#_Toc280930977)

[2.1 Regras 26](#_Toc280930978)

[2.2 Conceitos fundamentais 27](#_Toc280930979)

[2.2.1 Expected Value 27](#_Toc280930980)

[2.2.2 Pot Odds 28](#_Toc280930981)

[2.2.3 Implied Odds 29](#_Toc280930982)

[2.2.4 Ranges 30](#_Toc280930983)

[2.2.5 Posição 31](#_Toc280930984)

[2.2.6 Análise estatística 33](#_Toc280930985)

[3 AGENTES NO PÔQUER 35](#_Toc280930986)

[3.1 Propriedades de um agente para pôquer 36](#_Toc280930987)

[3.2 Com limite *versus* sem limite 36](#_Toc280930988)

[3.3 Avaliação de performance 38](#_Toc280930989)

[3.4 Arquiteturas e desenvolvimento histórico 39](#_Toc280930990)

[3.4.1 Sistemas baseado em regras 39](#_Toc280930991)

[3.4.1.1 SoarBot 40](#_Toc280930992)

[3.4.2 Sistemas baseados em fórmulas 41](#_Toc280930993)

[3.4.2.1 Loki 41](#_Toc280930994)

[3.4.3 Simulação 43](#_Toc280930995)

[3.4.3.1 Loki-2 e Poki 43](#_Toc280930996)

[3.4.4 Métodos baseados em teoria dos jogos 44](#_Toc280930997)

[3.4.4.1 PsOpti2 46](#_Toc280930998)

[3.4.4.2 GS1 48](#_Toc280930999)

[3.4.5 Busca adaptativa em árvores de jogo com informações imperfeitas 49](#_Toc280931000)

[3.4.5.1 Vexbot 49](#_Toc280931001)

[4 IMPLEMENTAÇÃO DO AGENTE 52](#_Toc280931002)

[4.1 API Meerkat 52](#_Toc280931003)

[4.2 Agente simples 53](#_Toc280931004)

[4.2.1 Arquitetura 54](#_Toc280931005)

[4.2.1.1 Pré-flop 54](#_Toc280931006)

[4.2.1.2 Pós-flop 55](#_Toc280931007)

[4.2.1.3 Outras considerações 57](#_Toc280931008)

[4.2.2 Principais elementos 57](#_Toc280931009)

[4.2.3 Limitações 58](#_Toc280931010)

[4.3 Agente completo 59](#_Toc280931011)

[4.3.1 Arquitetura 60](#_Toc280931012)

[4.3.1.1 Estratégias de aposta 60](#_Toc280931013)

[4.3.1.1.1 Pré-flop 62](#_Toc280931014)

[4.3.1.1.2 Pós-flop 66](#_Toc280931015)

[4.3.1.1.3 Tamanho das apostas 66](#_Toc280931016)

[4.3.1.2 Modelagem de oponentes 67](#_Toc280931017)

[4.3.1.2.1 Árvore de contexto 69](#_Toc280931018)

[4.3.2 Principais elementos 74](#_Toc280931019)

[4.3.3 Limitações 75](#_Toc280931020)

[5 TESTES E RESULTADOS 77](#_Toc280931021)

[5.1 Ambiente de testes 78](#_Toc280931022)

[5.1.1 PokerAcademy 79](#_Toc280931023)

[5.1.2 Open Meerkat Testbed 79](#_Toc280931024)

[5.1.3 Oponentes 80](#_Toc280931025)

[5.2 Agente simples 81](#_Toc280931026)

[5.2.1 Situação genérica 82](#_Toc280931027)

[5.2.2 ReduxBot *versus* ORenegado 83](#_Toc280931028)

[5.2.3 Oponente explorador 84](#_Toc280931029)

[5.2.4 Oponente previsível 85](#_Toc280931030)

[5.3 Agente completo 87](#_Toc280931031)

[5.3.1 Situação Genérica 87](#_Toc280931032)

[5.3.2 EssentialBot *versus* ReduxBot 89](#_Toc280931033)

[5.3.3 Embates individuais 90](#_Toc280931034)

[5.3.4 Agente previsível 90](#_Toc280931035)

[CONCLUSÃO 93](#_Toc280931036)

[REFERÊNCIAS BIBLIOGRÁFICAS 96](#_Toc280931037)

[ANEXOS 100](#_Toc280931038)

[ANEXO I – Históricos contra reduxbot 101](#_Toc280931039)

[ANEXO II – históricos contra agente previsível 104](#_Toc280931040)

[GLOSSÁRIO 108](#_Toc280931041)

introdução

Tão logo as linguagens de programação surgiram, a área de inteligência artificial atraiu uma grande parcela de interesse dos pesquisadores. O desafio de simular o comportamento racional em máquinas fez com que os estudiosos passassem centenas de horas desenvolvendo algoritmos voltados a esse propósito. Desde então, muito já foi conquistado e a realização de tarefas que antes pareciam possíveis apenas na ficção, como a existência de robôs limpadores em casas e automóveis que estacionam sozinhos, introduz um novo paradigma para o futuro.

Um domínio que sempre se sobressaiu na progressão histórica do campo da inteligência artificial é o de jogos. Desde o princípio, os pesquisadores obtiveram considerável sucesso em jogos como xadrez e damas – a ponto do supercomputador *Deep Blue*, anos depois, ser a primeira máquina a vencer um campeão mundial de xadrez (HSU, 2002). Os dois jogos anteriormente citados possuem algumas características em comum. Ambos são jogos determinísticos com informações perfeitas. Ou seja, é possível avaliar o estado completo do jogo apenas observando o tabuleiro, verificando quais peças estão presentes e sua respectiva posição na mesa. Adicionalmente, não há nenhum elemento surpresa ou qualquer tipo de aleatoriedade envolvida no desenvolvimento de estados futuros, garantindo uma quantidade finita de estados, mesmo que esta possa ser excessivamente volumosa.

Até recentemente, essa categoria de jogos era a única que recebia a devida atenção dos pesquisadores. Como possuem características mais intuitivas e menor complexidade de resolução, a solução dos mesmos utiliza uma abordagem semelhante. Para solucioná-los, geralmente aplicam-se algoritmos de força bruta baseados em *Minimax* e uma função de utilidade para avaliar as condições do jogo nos nodos finais, visto que não é possível simular a árvore inteira de jogo. O objetivo dessa função é verificar se a situação da partida em um determinado momento é favorável ao jogador (por exemplo, se as peças restantes estão dispostas adequadamente), possibilitando a escolha da jogada com melhor performance (RUSSEL; NORVIG, 1995).

Porém, nem todos os jogos contêm essas peculiaridades. Jogos com informações imperfeitas são assim chamados pois não se tem conhecimento de todos os dados necessários para a tomada de decisão, como as cartas do adversário ou a localização de suas peças, nos jogos de *bridge* e batalha naval, respectivamente. Assim, não é possível modelar o estado completo do jogo e os jogadores precisam praticar deduções sobre os dados inacessíveis. Já os jogos não determinísticos são aqueles que possuem o elemento chance, como o rolar de dados no gamão ou a distribuição de cartas no pôquer.

Alguns dos motivos que favorecem a utilização de jogos para pesquisas científicas são a existência de regras bem definidas, com metas claras e resultados mensuráveis, exigindo a utilização de estratégias complexas para se obter sucesso considerável (BILLINGS, 2006). Esses fatores permitem que os estudiosos foquem-se na parte relevante do problema, a criação das estratégias de jogo, e, conseqüentemente, evitem demasiada atenção aos detalhes de menor importância. Finalmente, essa arquitetura garante que a performance de um agente possa ser facilmente analisada utilizando métricas pré-definidas, como o número de partidas vencidas ou a quantidade de dinheiro ganha.

O pôquer, além das características acima, possui diversas propriedades que não estão presentes nos jogos costumeiramente estudados. Além da aleatoriedade presente no embaralhamento das cartas e da insuficiência de informação, que impediram que houvesse avanços na área durante longos anos, há, ainda, a existência de diversos agentes no mesmo jogo, a variância dos resultados no curto prazo e a necessidade de modelar os oponentes de acordo com suas características, lidando com a indução de erros através de blefes.

Adicionalmente, os estados ocultos do jogo são apenas parcialmente observáveis. Portanto, quando uma mão não chega até o *showdown*, ou seja, quando todos os oponentes desistem e um jogador ganha o pote sem contestação, não há necessidade de revelar as cartas privadas de cada jogador. Assim, os jogadores ficam impossibilitados de observar a maneira que os oponentes jogam suas mão e, dessa forma, encontrar tendências no estilo de jogo dos mesmos (MCCURLEY, 2009).

A modelagem de oponentes auxilia na identificação dessas tendências e permite com que seja possível prever o comportamento futuro de um adversário em situações distintas baseado em suas ações passadas. Para identificar padrões na forma de jogar dos oponentes, a solução proposta nesse trabalho usará um sistema de modelagem de oponentes através de árvores de contexto. O reconhecimento de padrões permite que um jogador adapte seu jogo com intuito de explorar as tendências do adversário, buscando por situações onde ele freqüentemente comete falhas.

As contribuições advindas de pesquisas na área de inteligência artificial para jogos, em especial os não determinísticos com informações imperfeitas, vão muito além do escopo da teoria dos jogos. Como muitos dos problemas encontrados nos jogos são similares aos situados no mundo real, as soluções podem ser adaptadas e utilizadas em diferentes situações. Por exemplo, o sistema de busca heurística criado pela Universidade de Alberta é também utilizado para encontrar a melhor localização para a instalação de bueiros nas cidades (JOHANSON, 2007). Problemas financeiros, militares, meteorológicos e de todas as áreas que possuem características semelhantes às encontradas em jogos podem também se beneficiar de eventuais contribuições dessa e outras pesquisas.

Entre as diversas variedades de pôquer existentes, esse projeto se focará no *Texas Hold’em*. Hoje, esse é o estilo de pôquer mais jogado no mundo e, conseqüentemente, alvo da maior parte dos pesquisadores. Sendo assim, há uma bibliografia extensa que pode amparar o desenvolvimento do projeto, além da existência de grupos de pesquisas específicos sobre o tema. Apesar disso, o trabalho focar-se-á na modalidade sem limite de apostas do jogo, enquanto a maioria das publicações refere-se ao jogo com limite. Assim, a aplicação de algumas técnicas relatadas em outros trabalhos será condicionada a viabilidade das mesmas para o novo domínio.

Portanto, esse projeto visa à criação de um agente racional combinando diferentes técnicas computacionais e analisando o jogo de um ponto de vista cognitivo. Pretende-se introduzir estratégias utilizadas pelos melhores jogadores de pôquer, como a leitura de mãos com o conceito de *ranges*, e técnicas de inteligência artificial já comprovadamente eficientes, como a modelagem de oponentes através de árvores de contexto. Para desenvolver e testar o agente, será utilizado o programa *Poker Academy*, criado pela Universidade de Alberta. Ele permite a implementação de agentes utilizando sua API *Meerkat*, que disponibiliza uma série de rotinas para a interação com o programa. Depois, no software, é possível importar os agentes criados com a API e testá-los contra as melhores inteligências artificiais desenvolvidos pelo *Computer Poker Research Group* da Universidade de Alberta, uma das instituições mais reconhecidas no campo das pesquisas de agentes para pôquer.

No primeiro capítulo, o conceito de agentes é abordado. Aspectos importantes para a compreensão do tema são esclarecidos, como a distinção entre arquiteturas reativas, deliberativas e híbridas. As características desejáveis em um agente são detalhadas, sintetizando a importância e o alcance das mesmas. Após, a propriedade racional dessas entidades é discutida. Na segunda parte do trabalho, o referencial teórico sobre o *Texas Hold’em* é aprofundado, pormenorizando as regras do jogo e as ideias essenciais presentes no agente proposto. Já no terceiro capítulo, o conteúdo das seções anteriores é integrado, conciliando os fundamentos de agentes e pôquer em um único tópico. Analisa-se, também, o desenvolvimento histórico dos agentes para pôquer, refletindo sobre as importantes contribuições de cada abordagem. No quarto capítulo, todos esses conceitos são então absorvidos e empregados no desenvolvimento de dois agentes distintos, com premissas de funcionamento únicas mas que compartilham a mesma arquitetura essencial. Por fim, na última seção, os agentes construídos são testados sob circunstâncias peculiares e genéricas, objetivando avaliar sua performance e sua conformidade com outros conceitos discutidos durante o trabalho.

# agentes

Como em muitos outros conceitos teóricos criados pelo ser humano, os agentes não possuem uma definição concreta e amplamente aceita pela comunidade que os estuda. Diferentes autores costumam destacar diferentes características de seu funcionamento, enfatizando aspectos distintos como a autonomia ou a racionalidade dos mesmos. Dentre as definições existentes, Maes (1995, p. 108) diz:

“Agentes autônomos são sistemas computacionais que habitam algum ambiente complexo e dinâmico, percebem e agem dentro desse ambiente e, dessa forma, realizam um conjunto de metas e tarefas cujas quais foram desenhados para resolver.” (Tradução nossa)

Contudo, embora não haja concordância na definição do termo, há assentimento de que um agente deve ser capaz de perceber corretamente o ambiente no qual está inserido através de sensores e atuar nesse ambiente com atuadores (RUSSEL; NORVIG, 1995). A Figura 1.1 ilustra essa arquitetura.

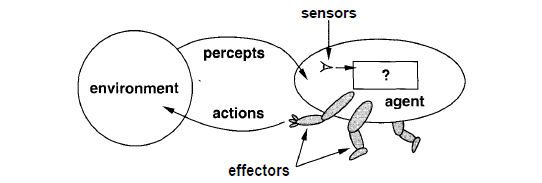


Figura 1 Arquitetura de um agente

Fonte: RUSSEL e NORVIG, 1995

Essa estrutura é útil de um ponto de vista didático pois abstrai a complexidade do agente, visualizando-o somente como uma entidade que interage com um determinado ambiente e omitindo os detalhes de implementação. Dessa forma, visualiza-se que todo agente, independente do domínio que estiver alocado, possuíra algumas propriedades equivalentes.

Segundo Wooldridge e Jennings (1995), existem duas maneiras gerais no qual o termo agente é usado. A primeira associa um sistema computacional a atributos como autonomia, habilidade social, reatividade e pró-atividade, estabelecendo uma noção fraca de agência. A segunda possui um significado mais forte e define os agentes de forma mais específica, como sistemas computacionais que, além de possuírem as características já citadas, são idealizados ou implementados utilizando conceitos usualmente aplicados em humanos, como noções de conhecimento, metas e crenças.

## Características de um agente

Entre as características desejáveis em um agente, apresentar-se-á aquelas que possuem destacada importância no âmbito desse trabalho. A lista abaixo denomina e detalha alguns desses atributos.

* Autonomia: capacidade do agente de tomar iniciativa e obter um grau de controle elevado sobre suas próprias ações (ETZIONI; WELD, 1995).
* Reatividade: o agente percebe o ambiente no qual está inserido e responde rapidamente as mudanças que ocorrem no mesmo (WOOLDRIDGE; JENNING, 1995).
* Preditividade: um agente é preditivo se sua visão do ambiente é suficientemente precisa para que ele possa predizer como realizar seus objetivos (GOODWIN, 1993).
* Racionalidade: o agente age de uma maneira que, conforme suas deduções, atingirá suas metas e objetivos (GOODWIN, 1993).
* Sociabilidade: propriedade de um agente capaz de interagir com outros agentes para realizar seus objetivos ou auxiliar na resolução dos problemas de outro agente.

## Percepção

Para que um agente obtenha êxito em seus objetivos, é vital que ele seja capaz de perceber o ambiente no qual está inserido com precisão. Um agente situado no mundo físico pode precisar de sensores de luz, temperatura, imagem e som para obter as informações que serão necessárias para a tomada de decisão. Já um agente *software,* ou *softbot*, pode utilizar uma interface gráfica para interagir com o usuário, o reconhecimento de imagens e texto, a comunicação com outros agentes ou até mesmo a internet para essa mesma função.

Independente dos meios utilizados para a modelagem do ambiente, o agente deve ser capaz de adquirir todas as informações julgadas necessárias e identificar corretamente as diferentes situações em que o ambiente pode se encontrar. Depois, essas informações são processadas e o agente deve optar pela melhor ação disponível, de acordo com suas metas. Caso a percepção do ambiente seja realizada de maneira incorreta, o agente, devido à insuficiência de informação, pode tomar ações que, embora sejam racionais considerando os dados acessíveis, não representam o comportamento esperado.

Portanto, a habilidade de distinguir fielmente o ambiente é crucial do ponto de vista de desempenho e tem grande influência no sucesso geral do sistema. Além disso, uma falha perceptiva pode causar danos catastróficos dependendo das funções que o agente deve desempenhar, como, por exemplo, no caso de um robô que auxilia a movimentação de deficientes visuais.

## Arquiteturas

Costumeiramente, o termo arquitetura se refere à estrutura física de algo. A arquitetura de Von Neumann, por exemplo, define um modelo para computadores digitais baseado em cinco componentes (memória, unidade aritmética e lógica, unidade central de processamento, entradas e saídas). Em contraposição, as arquiteturas de agentes caracterizam, além da estrutura física dos mesmos, a maneira como as informações adquiridas devem ser processadas e quais dados são relevantes para serem armazenados internamente. Nas seções seguintes, formalizar-se-á alguma das arquiteturas de agentes utilizadas pelos pesquisadores na resolução de problemas.

### Arquiteturas reativas

Os agentes reativos são conhecidos por sua estrutura simples e que, em muitos casos, prova-se eficaz. O sistema de tomada de decisão é baseado apenas em regras condicionais, limitando a racionalização requerida pelo agente para pouca ou nenhuma. Não há qualquer tipo de memória presente e, por consequência, o agente não mantem controle do estado atual do ambiente. Suas ações são desencadeadas somente por entradas fornecidas pelos sensores. A Figura 2 ilustra uma arquitetura reativa.

Como grande parte das tarefas executadas por humanos também não requer um considerável grau de racionalização, sendo freqüentemente realizadas de forma rotineira, uma arquitetura baseada em regras pode ser pertinente. A inexistência de memória é explicada pelo fato de não haver representação interna do estado do ambiente, evitando qualquer tipo de planejamento futuro. Assim, o sistema só precisa ser alterado quando se deseja que o agente desempenhe novas funções. Isso é atingido criando novas regras condicionais ou alterando as existentes.

Apesar da sua simplicidade, os agentes reativos podem realizar ações que requereriam um alto grau de racionalização em outras arquiteturas. Steels (1990) descreve um sistema de exploração do planeta Marte utilizando um grande número de agentes com arquiteturas *subsumption*, uma ramificação das arquiteturas reativas, e que poderia atingir um nível de desempenho quase perfeito.

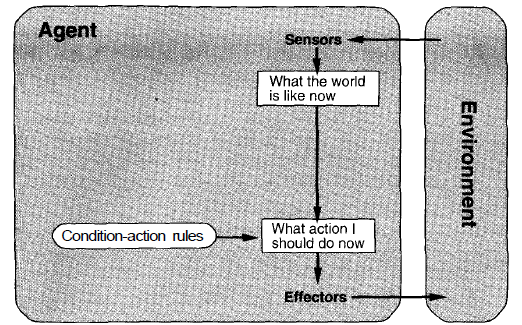


Figura 2 Arquitetura reativa

Fonte: Russel e Norvig, 1995

Uma das motivações no desenvolvimento das arquiteturas reativas é a impraticabilidade da modelagem interna do ambiente. Agre e Chapman (1987) afirmam que embora a IA constantemente utilize o conceito de planejamento para a tomada de decisão, ele é irrealizável em situações realísticas devido à alta dificuldade em coletar todas as informações necessárias para modelar corretamente o estado do ambiente. Brooks (1991) adiciona que essas representações perfeitas do ambiente, além de inexequíveis, não são necessárias para se atingir um bom nível de desempenho. Essa visão é especialmente verdadeira em ambientes onde as entradas do sistema são muito complexas e a modelagem do mesmo tem duração elevada, como pode ocorrer tanto em domínios reais quanto virtuais.

Por fim, um agente que não realize planejamento garante flexibilidade na mudança de comportamento caso novas situações ocorram durante a execução de uma ação, haja vista que ele não será compelido a executar um plano contínuo de ações até seu término, como ocorre em outras arquiteturas. Essencialmente, essa propriedade é útil em ambientes incertos, onde a natureza imprevisível dos mesmos pode freqüentemente alterar as condições de trabalho do agente e, algumas vezes, necessitar de uma mudança de ação imediata para que o agente continue a alcançar os objetivos propostos.

### Arquiteturas deliberativas

A ideia de comportamento inteligente que as arquiteturas reativas promovem, baseada somente em regras condicionais, não satisfaz completamente aquilo que pressupõe-se como raciocínio lógico. Os agentes deliberativos, em oposição, simulam mais fielmente o que imagina-se ser o processo de racionalização dos seres humanos em face de decisões difíceis. Wooldridge e Jennings (1995, p. 24) definem um agente ou arquitetura deliberativa da seguinte maneira:

“[...] aquela que contem um modelo simbólico e explicitamente representado do ambiente, cujas decisões (por exemplo, de qual ação executar) são realizadas através de raciocínio lógico (ou, pelo menos, pseudo-lógico) baseado em pareamento de padrões e manipulação simbólica.” (Tradução nossa)

Por essa definição, percebe-se que as arquiteturas deliberativas armazenam internamente uma representação do ambiente e continuamente manipulam os seus estados futuros. Os agentes deliberativos são freqüentemente baseados em planejamento e possuem uma noção mais concreta dos objetivos que devem realizar. O planejamento analisa, além das possíveis ações e seus resultados, as condições atuais do habitat e as prováveis ações dos outros agentes existentes, considerando suas consequências no ambiente.

Muitos dos protótipos iniciais de agentes deliberativos possuíam um planejador como elemento crucial do sistema. O planejador tem o objetivo de determinar a sequência de ações que deve ser realizada para alcançar os objetivos propostos, considerando o estado atual do ambiente e o estado desejado. Em seu sistema de combate a incêndios, Cohen (1989) implementa um planejador que seleciona o plano de ações a ser tomado entre diversos planos pré-determinados e, então, comunica-se com agentes subordinados para delegar as ordens de combate ao alastramento do fogo.

Na Figura 3, que demonstra uma estrutura deliberativa contendo um subsistema de planejamento como componente principal, a arquitetura do agente incorpora uma base de crenças cujos planos devem submeter-se. Isso é, as ações do sistema sempre deverão estar de acordo com suas crenças. A existência de um planejador pode auxiliar na performance do agente diminuindo a quantidade de tempo de processamento na procura de uma solução, visto que as possíveis ações resultantes são limitadas pela quantidade de planos que se encaixem com a situação, e não vice-versa. Ou seja, as situações onde não existem planos correspondentes são rapidamente eliminadas sem que haja necessidade de uma busca extensiva de quais ações são possíveis no estado atual.

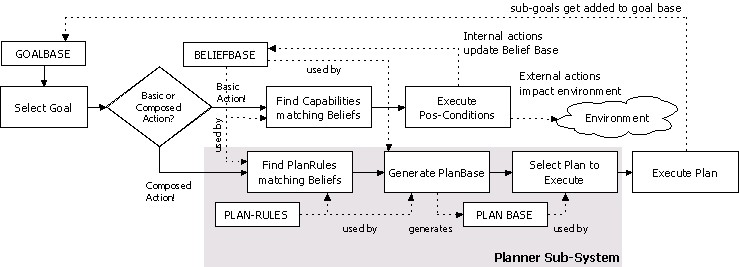


Figura 3 Arquitetura deliberativa

Fonte: Koch, 2005

Como a abordagem deliberativa foca-se na representação simbólica do ambiente e na manipulação dessas informações para obter e ajustar o comportamento adequado do agente, existem duas questões relevantes que devem ser consideradas: o problema de transdução e o problema de representação/racionalização. O problema de transdução refere-se à dificuldade em traduzir o ambiente para uma descrição simbólica apropriada em tempo hábil. Já o problema de representação/racionalização questiona como representar simbolicamente as informações contidas em um ambiente real e complexo, também indagando como fazer com que os agentes utilizem essas informações na sua racionalização e cheguem a uma decisão em tempo útil. Esses dois problemas contribuíram amplamente para pesquisas em tópicos como reconhecimento de fala, aprendizado de máquina, planejamento automático e representação de conhecimento, mas ainda assim estão longe de serem resolvidos (WOOLDRIDGE; JENNINGS, 1995).

### Arquiteturas híbridas

Devido à grande controvérsia que as arquiteturas reativas e deliberativas provocavam entre seus adeptos, e considerando os limites de aplicação de cada uma delas, alguns pesquisadores sugeriram a criação de arquiteturas híbridas. Como exemplo desse pensamento, Jennings, Sycara e Wooldridge (1998) asseguram que nem arquiteturas puramente deliberativas ou arquiteturas puramente reativas são apropriadas para a maioria dos problemas. A abordagem híbrida é uma tentativa de combinar as vantagens das arquiteturas deliberativas e reativas, unificando componentes de ambas, como a representação simbólica do ambiente e a capacidade de planejamento das arquiteturas deliberativas com a simplicidade de decisão das arquiteturas reativas, em um único sistema. Geralmente, os sistemas híbridos são compostos de múltiplas camadas. A Figura 4 representa uma abstração de uma arquitetura híbrida em camadas.

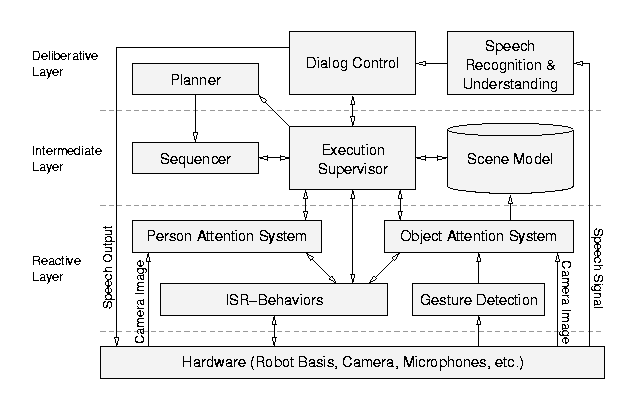


Figura 4 Arquitetura híbrida

Fonte: Haasch et al (2004)

Percebe-se que a estrutura acima possui três camadas organizadas de forma hierárquica. A primeira camada, que age diretamente sobre as entradas do sistema, é baseada em componentes reativos. A camada intermediária funciona como um *middleware* e possibilita a representação simbólica do ambiente. Finalmente, a última camada, baseada em características deliberativas, possui um planejador de ações. Essa arquitetura é bastante semelhante com a exemplificada em Jennings, Sycara e Wooldridge (1998).

Outra possível forma de organizar um agente híbrido é com a utilização de camadas horizontais. Na estrutura detalhada acima, as informações são filtradas e depois passadas sequencialmente entre os componentes. Com camadas horizontais, no entanto, as camadas têm acesso às entradas e saídas na sua totalidade e chegam a conclusões baseadas nos dados completos, sendo necessário um mediador para preterir entre as decisões de cada uma delas e escolher a ação correta.

Nos últimos anos, as arquiteturas híbridas tem recebido uma boa parcela de atenção dos pesquisadores pois a flexibilidade e o dinamismo da estrutura mostra-se eficiente na resolução de problemas em diversos domínios. Ainda assim, o alto grau de complexidade envolvido na unificação com clareza entre componentes deliberativos e reativos inibe a maior disseminação dessa abordagem (WOOLDRIDGE, 1995). Em escalar menor, problemas de comunicação entre as camadas também coíbem a utilização de agentes híbridos.

## Agentes racionais

No contexto da teoria dos jogos, uma decisão racional é aquela que possui o melhor retorno possível considerando todas as opções de ações disponíveis. Esse retorno é muitas vezes especificado por uma função de utilidade U, onde uma estratégia x terá preferência sobre y apenas se U(x) é maior ou igual a U(y) (OSBORNE; RUBINSTEIN, 1994). Portanto, um agente racional é aquele que toma a ação de maior sucesso conforme sua percepção do ambiente – considerando que o êxito dessa ação seja avaliado através de métricas de performance bem elaboradas.

Caso exista uma ação com retorno esperado superior a tomada pelo agente, isso não implica em sua irracionalidade, contando que ele não esteja ciente da mesma (RUSSEL; NORVIG, 1995). Ou seja, supondo que um jogador sempre escolha papel no jogo de pedra, papel e tesoura, um agente recém-inserido nesse ambiente não pode ser considerado irracional se não escolher a contra estratégia correta (pedra), visto que não possui conhecimento histórico das tendências do adversário.

Tipicamente, os agentes racionais não possuem uma arquitetura definida. Ainda assim, é necessário que ele seja capaz de prever e simular o resultado de diferentes ações para definir qual será a melhor delas. Ou seja, uma arquitetura puramente reativa torna-se impraticável devido à ausência de memória interna para a representação simbólica do ambiente, impossibilitando simulações extensivas dos diferentes cursos de ações disponíveis. Portanto, para que se possa determinar a melhor decisão, é necessário que a estrutura do agente possua, ao menos, um componente deliberativo.

### Avaliando a melhor decisão

Para uma avaliação correta da melhor decisão, é fundamental que as metas e preferências do agente estejam bem definidas. Essa questão é amplamente estrutural e deve ser analisada desde o início do projeto. Com isso, é possível estabelecer preferência entre duas ações distintas facilmente, considerando as possíveis consequências destas ações e a constante busca por condições de jogo mais propícias ao agente.

No pôquer, a melhor decisão será geralmente aquela que possui o maior retorno financeiro imediato. Eventualmente, haverá situações onde é preferível a escolha de uma jogada com expectativa menor contando que ela seja capaz de induzir os oponentes a erros no futuro. Por exemplo, se um jogador realiza um blefe mal sucedido no início de uma partida, é muito provável que seus adversários estejam aptos a pagá-lo futuramente com mãos mais fracas. Dessa maneira, mesmo que a jogada tenha obtido um retorno negativo no curto prazo, ela induzirá o oponente a falsas suposições e certamente aumentará a expectativa das jogadas posteriores.

Diante disso, o processo de escolha de ação de um agente racional não deve levar em consideração somente o retorno esperado de uma jogada, mas, também, deve analisar suas metas e preferências, sua percepção do ambiente, suas ações disponíveis, sua avaliação do sucesso dessas ações e como cada uma delas influenciará nas condições de jogo e nas decisões futuras dos oponentes.

## Aplicações de agentes

O campo de aplicação dos agentes é caracterizado pela sua amplitude. A aplicabilidade dos agentes tanto em domínios virtuais quanto reais permite que eles sejam considerados na resolução das mais variadas classes de problemas. As seções seguintes exemplificam três soluções agenciais propostas para diferentes áreas, detalhando brevemente seu funcionamento.

### Controle de tráfico aéreo

O sistema proposto por Tumer e Agogino (2007) utiliza uma abordagem com múltiplos agentes para reduzir os congestionamentos aéreos em até sessenta e sete por cento em relação aos métodos comumente utilizados pela indústria aeronáutica. Nele, cada agente é responsável pelo monitoramento de uma área específica do espaço aéreo e busca manter uma determinada distância entre os aviões que passam nessa região, solicitando, quando necessário, mudanças de velocidade aos pilotos para evitar congestionamentos.

### Filtragem de informação

O excesso de informação, especialmente em domínios virtuais onde há uma imensa quantidade de dados a disposição, pode causar desorientação e prejudicar a experiência geral do usuário. Maes (1994 apud JENNINGS; WOOLDRIDGE, 1998) propôs o sistema de filtragem de notícias Newt, baseado no conceito de agentes, e que mantem um controle interno dos interesses do usuário. O programa inicialmente questiona o utilizador sobre o interesse do mesmo na leitura de determinados artigos e, de acordo com as respostas, modela suas preferências para posteriormente recomendar outras notícias que possam ser relevantes.

Adicionalmente, Maes (1994 apud JENNINGS; WOOLDRIDGE, 1998) utiliza uma abordagem semelhante no sistema de filtragem de e-mails Maxims. Nesse caso, porém, o agente não interage diretamente com o usuário para compreender seus interesses. Ao contrário, esse objetivo é atingido apenas monitorando a utilização normal do serviço de e-mail e predizendo as decisões do usuário. Quando o sistema passa a obter uma boa taxa de precisão em suas deduções, ele começa a sugerir planos de ações comumente realizados para aquele tipo de mensagem. Dessa forma, o programa é capaz de aprender a priorizar, apagar, arquivar e até repassar os e-mails recebidos baseado nas tendências do usuário.

### Comércio Eletrônico

Nos dias de hoje, grandes sites de comércio online como eBay e Amazon já conseguem antecipar as necessidades de seus clientes e constantemente recomendam produtos que as satisfaçam. Guttman, Moukas e Maes (2000) analisam diferentes aplicações de agentes no comércio eletrônico e as posiciona de acordo com seu papel nos modelos de comportamento do consumidor. Essas aplicações serviriam como mediadores no processo de compra com intuito de auxiliar o consumidor na procura do melhor produto ou oferta. Entre as funções que estes programas desempenham, destacam-se a identificação das necessidades do consumidor, a procura pelo melhor produto da categoria desejada, a seleção do melhor vendedor entre os que anunciam um determinado produto, o processo de negociação com o vendedor, a efetivação da compra e a avaliação do serviço prestado.

Essas aplicações representam um avanço em relação ao atual uso corrente de agentes no processo de compra eletrônica, que situa-se apenas na recomendação de produtos, e sua utilização impulsionaria o desenvolvimento de programas semelhantes visando proteger o consumidor. Por outro lado, o emprego de tais aplicações infere em uma questão delicada de confiança entre o usuário e a máquina, sendo que alguns paradigmas precisariam ser quebrados antes da implantação em grande escala de programas desse tipo.

# TEXAS HOLD’EM PÔQUER

O Texas Hold’em é a variedade de pôquer mais popular na atualidade. Jogado por centenas de milhares de pessoas a cada dia, seja em cassinos, em casa ou até via internet, o jogo combina a matemática com a psicologia e representa um desafio para todos os pesquisadores da área de inteligência artificial. Como na maioria dos tipos de pôquer, o Texas Hold’em pode ser jogado com ou sem limite de apostas. O jogo com limite de apostas define um valor fixo para cada aposta dependendo da rodada em que se encontra. Na espécie sem limite de apostas, os jogadores recebem apenas um valor mínimo para cada aposta e podem escolher qualquer valor superior a este se desejam apostar ou aumentar a aposta de um adversário.

Inicialmente, o Texas Hold’em com limite de apostas possuía predileção entre os jogadores e grande parte da bibliografia sobre o tema se refere a essa variedade do jogo. Com o advento da transmissão televisionada dos campeonatos de pôquer, tipicamente jogados sem limite de apostas, a preferência dos jogadores passou a ser dos jogos sem limite. As pesquisas sobre o tema ainda encontram-se em processo de transição e maioria delas ainda utiliza o jogo com limite de apostas como domínio de estudo. O agente detalhado nesse trabalho é projetado para jogos sem limite de apostas, por opção do autor.

Esse capítulo tem como objetivo definir as regras do Texas Hold’em e detalhar alguns dos conceitos fundamentais que servirão como base de conhecimento do agente proposto. Depois, na próxima parte, o desenvolvimento histórico dos agentes para pôquer é analisado cronologicamente, comparando as diferentes arquiteturas e expondo as vantagens e desvantagens de cada uma delas.

## Regras

Primeiramente, os dois jogadores a esquerda do *dealer* pagam duas apostas forçadas chamadas de *blinds* (cegas). Depois disso, cada jogador recebe duas cartas privadas e há uma primeira rodada de apostas, conhecida como *pré-flop*. Em todas as rodadas de apostas, o jogador pode escolher entre passar, pedir mesa/pagar ou apostar. O valor máximo de cada aposta é restrito ao tipo de jogo, sendo possível apostar qualquer valor em jogos sem limite.

Ao fim do *pré-flop*, quando os participantes já decidiram suas ações, três cartas são distribuídas na mesa e são utilizadas por todos os jogadores restantes. Essas três cartas são chamadas de *flop*. Mais uma rodada de apostas acontece e uma quarta carta comunitária, conhecida como *turn*, é distribuída. Após o término da rodada, uma quinta e última carta comunitária vem à mesa, o *river*. Há, então, uma última rodada de apostas e a mão vai para o *showdown*. Nele, os jogadores remanescentes mostram suas cartas privadas e a melhor mão de cinco cartas, de acordo com os *rankings* do jogo, vence. Se em qualquer momento da partida restar apenas um jogador, isto é, quando todos os seus oponentes desistem, a mão é finalizada e o *pot* é cedido ao jogador em questão.

## Conceitos fundamentais

No pôquer, e em qualquer outro jogo de apostas, o objetivo do jogador será sempre escolher a ação com o maior retorno esperado. Determinar a lucratividade de um ato, porém, é uma tarefa que depende de diversos fatores, tais como o potencial ganho e a probabilidade de vitória. Os conceitos que serão apresentados nas seções seguintes servem como ferramentas de análise no jogo de pôquer e fazem parte dos fundamentos básicos que constituem o agente proposto neste trabalho.

### Expected Value

*Expected Value* é a expectativa de retorno médio de uma jogada. É um conceito de suma importância e permite que um jogador avalie a lucratividade média de suas jogadas (THORP, 1985). Baseia-se em um cálculo matemático incluindo a probabilidade de vitória, a probabilidade de derrota, o valor que pode ser perdido e o valor que pode ser ganho. A Figura 5 ilustra a fórmula básica de EV.



Figura 5 Fórmula de EV

Fonte: Elaborado pelo autor

Para demonstrar o uso da fórmula, pode-se imaginar um jogo de dados onde um jogador ganha quatro dólares toda vez que o número seis é sorteado e paga um dólar para qualquer outro número. Nesse caso, conforme ilustra a Figura 6, sua expectativa será de menos dezesseis centavos. Mesmo que os únicos resultados individuais dessa aposta sejam o lucro de quatro dólares ou o prejuízo de um dólar, ela resultará, na média, em um retorno negativo de dezesseis centavos por jogada. Por óbvio, conclui-se que a expectativa da aposta será sempre a mesma independente de qualquer resultado individual.

Nesse sentido, a expectativa serve apenas como referência ao sucesso de uma jogada já que, devido à presença de elementos estocásticos no jogo de pôquer e das flutuações que o não determinismo representa, as variações de resultado podem ser significativas no curto prazo e só tendem a convergir após uma amostragem considerável. Ainda assim, mesmo que a existência do fator sorte possua alguma influência nas saídas do jogo, ele tende a ser sobreposto pela habilidade dos jogadores e pode ser desconsiderado no aspecto geral de uma partida (CROSON; FISHMAN; POPE, 2008). Por outro lado, a aleatoriedade permite que jogadores fracos permaneçam jogando quando falsamente iludem-se que seus prejuízos são causados pelo azar.



Figura 6 EV do jogo de dados

Fonte: Elaborado pelo autor

### Pot Odds

Basicamente, o conceito de *pot odds* cria uma relação entre a quantia no pote, a aposta do oponente e o quanto precisa ser pago, sendo possível identificar a parcela de tempo que um jogador precisa vencer para poder pagar uma aposta lucrativamente. Por exemplo, se após todas as cartas da mesa houverem sido distribuídas há cinquenta dólares no pote e o adversário aposta vinte e cinco dólares, o jogador precisará pagar vinte e cinco dólares para poder ganhar os setenta e cinco dólares que estão em jogo. Utilizando o sistema de probabilidade americano, pode-se dizer que há uma proporção de 3 – 1 (três para um) entre o que pode ser ganho e o que deve ser pago. Essa proporção de 3 – 1 (três para um) será o *pot odds* do jogador em questão. A Figura 7 demonstra o cálculo da quantidade de tempo que será necessário vencer para que o *call* tenha uma expectativa positiva no exemplo acima.

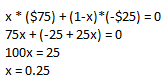


Figura 7 Lucratividade do *call*

Fonte: Elaborado pelo autor

Portanto, com pot odds de 3 – 1 (três para um), é possível esperar lucro se o jogador vencer uma quantidade de tempo superior a vinte e cinco por cento. Conclusivamente, pode-se dizer que se a porcentagem do valor que o jogador contribui para o pot é maior do que sua probabilidade de vitória, ele automaticamente perderá dinheiro no longo prazo. O inverso também é verdadeiro, visto que se o jogador contribuir com uma porcentagem menor que suas chances de vitórias, ele lucrará.

Assim sendo, esse conceito pode ser utilizado em dois tipos de situação: na última rodada, quando não serão distribuídas mais cartas, ou nas rodadas anteriores, quando ainda existe necessidade de distribuição de cartas até o fim do jogo. No primeiro caso, segundo Sklansky (1994), a decisão é tomada estimando as chances de o oponente estar apostando com uma mão inferior e se essas chances são melhores ou piores que o *pot odds* atual. Porém, no segundo caso, quando o jogo não se encontra na rodada final e outras cartas ainda serão distribuídas, o uso desse conceito é menos frequente visto que os estados futuros do jogo podem alterar a probabilidade de vitória de uma mão e/ou a quantidade de dinheiro que será paga até o *showdown*. Para tanto, o conceito de *implied odds*, que será aprofundado na seção seguinte, é utilizado.

### Implied Odds

Quando existem rodadas futuras, a possibilidade do jogador ser compelido a pagar mais apostas até o fim da mão é muito grande. Negligenciar essa variável e tomar decisões baseando-se apenas no *pot odds*, que considera somente o estado atual do jogo, pode tornar-se custoso a um jogador. A ideia principal do conceito de *implied odds* é analisar, além do estado atual, os possíveis acontecimentos nas rodadas futuras. Sklansky (1994) define *implied odds* como a relação entre o valor total a ser ganho com o custo atual de uma aposta. O exemplo a seguir ilustra com mais clareza esse conceito.

Na terceira rodada de um jogo de pôquer, o *turn*, um jogador aposta quinze dólares em um pote de quarenta dólares. O seu oponente possui uma queda para *flush*, isto é, quatro cartas do mesmo naipe, e precisa pagar a aposta do adversário para ver a última carta na mesa. Das quarenta e seis cartas desconhecidas no baralho, existem nove que completam sua mão e o fazem vencer. Assim, suas chances de vitória são de 37 – 9 (trinta e sete para nove) ou, simplificando, 4.1 – 1 (quatro ponto um para um). O *pot odds* do jogador é de 55 – 15 (cinquenta e cinco para quinze) ou 3.7 – 1 (três ponto sete para um). Ou seja, o jogador possui 19,6% (dezenove vírgula seis por cento) de chances de vencer e precisa contribuir com 21,2% (vinte e um vírgula dois por cento) do *pot*.

Analisando apenas essas variáveis, a decisão correta do jogador em questão seria desistir da mão pois está contribuindo com uma porcentagem maior do *pot* do que suas chances de vitória. Isso, invariavelmente, resultaria em prejuízo no longo prazo caso continuasse. Porém, como ainda há mais uma rodada de apostas no *river*, o jogador pode ganhar um valor maior do que o atualmente no pote se completar sua mão. Caso esse valor seja grande o suficiente para que o jogador passe a contribuir com uma porcentagem menor do *pot* do que suas chances de vitória, a decisão correta passa a ser pagar a aposta do oponente. A Figura 8 demonstra o valor mínimo que o jogador precisa vencer para continuar com a mão.

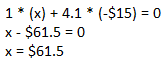


Figura 8 Valor mínimo para pagar lucrativamente

Fonte: Elaborado pelo autor

Compreende-se então que, no exemplo dado, o jogador precisaria ganhar sessenta e um dólares e cinquenta centavos para pagar a aposta de seu adversário no *turn* lucrativamente. Considerando que o pote já possui cinquenta e cinco dólares, seria necessário angariar apenas seis dólares e cinquenta centavos no *river.* Avaliar corretamente se esse valor será obtido é uma questão de julgamento.

### Ranges

O conceito de ranges é, provavelmente, um dos mais revolucionários do pôquer moderno. Apesar de já estar presente no processo de tomada de decisão dos jogadores desde antigamente, ele só foi formalizado com a introdução dos softwares de análise estatística dos jogadores no pôquer online. Essencialmente, uma range consiste em conjunto com todas as possíveis mãos do adversário em uma determinada situação. Ou seja, para cada ação de um jogador, existe um conjunto finito de mãos cujas quais ele realizaria essa mesma decisão. Essa faixa de mãos não é estática e deve ser reavaliada a cada ação tomada, estreitando o número de mãos possíveis na medida em que as rodadas progridem. O exemplo abaixo clarifica a explicação.

Em uma mesa com seis jogadores, todos os jogadores passam até Bob, sentado na última posição, que aumenta para três vezes o valor da cega com duas rainhas. O *small blind* passa e o *big blind*, Matt, apenas paga. Bob, conhecendo o jogo de seu adversário, sabe que ele é um jogador que joga poucas mãos e costuma pagar apostas nas cegas apenas com pares médios/baixos e mãos com duas cartas altas. O *flop* é T♥ 5♣ 2♠ e Matt pede mesa. Bob aposta 2/3 do valor do pote e Matt rapidamente paga.

Na sua primeira avaliação, quando apenas foi pago no *pré-flop*, Bob definiu que a range do oponente consistia de pares médios/baixos e cartas altas. Quando Matt paga novamente no *flop*, é seguro dizer que ele, sendo um jogador competente, dificilmente continuaria sem um par. Então, sua *range* se estreita para pares médios e cartas altas com um T, como AT, KT, QT e JT. Agora, com uma melhor noção do que seu adversário pode ter, Bob pode continuar jogando de forma ideal nessa e nas próximas rodadas.

No *turn*, independente da carta que for distribuída, Bob saberá exatamente como essa carta se relacionará com a *range* do oponente. Caso seja um rei, por exemplo, a única mão na *range* do adversário que se beneficiaria é o KT. Intuitivamente, percebe-se que essa determinada mão representa apenas uma pequena parcela da *range* de Matt e, portanto, não precisa ser temida irracionalmente. Então, mesmo com uma carta maior que seu par na mesa, Bob poderá continuar apostando confiantemente já que ainda existem diversas mãos inferiores a sua na *range* do adversário e que pagariam mais apostas.

Nesse exemplo, pode-se perceber a utilidade desse conceito e sua influência nas decisões dos jogadores. Na prática, porém, é impossível definir a maneira que os adversários jogariam cada uma das possíveis mãos em todas as distintas situações que podem ocorrer. Assim, as avaliações de faixa de mãos são geralmente baseadas em deduções, tendências observadas, análises estatísticas ou no senso comum.

### Posição

Um importante aspecto estratégico do *Texas Hold’em* é a posição de cada jogador em relação às apostas cegas, que define a ordem sequencial de cada jogador. No *pré-flop*, o primeiro jogador a agir será aquele que estiver sentado diretamente à esquerda do que pagou a cega maior (UTG). Nas outras rodadas, as apostas sempre se iniciam com o jogador que pagou a cega menor (SB) e, como os turnos seguem em sentido horário, fecham no jogador diretamente a direita do mesmo (BTN). Geralmente, os jogadores são referidos pela posição que estão sentados na exemplificação de uma mão. A Figura 9 ilustra a nomenclatura de cada posição em um jogo com seis jogadores.

Figura 2.4 Posições com seis jogadores

Conforme Miller, Sklansky e Malmuth (2004), a posição de um jogador é o segundo componente mais importante em uma mão de pôquer, a parte de suas cartas e as cartas dos adversários. Essa visão é compartilhada por Sklansky (1994), que afirma “a posição de um jogador na sequência de apostas é um aspecto importante, porém subestimado”. Ainda, Sklansky (1994) ressalta que a posição é um elemento chave e afeta o raciocínio de qualquer jogada no pôquer. Analisando a Figura 9, pode-se perceber que quanto mais distante das cegas (no sentido horário), mais turnos o jogador deve esperar para agir. A lista a seguir apresenta algumas das vantagens de uma boa posição.

* Informação: o pôquer é um jogo de informação, onde as decisões são tomadas com base nos dados disponíveis sobre a mão atual e os jogadores nela presentes. Portanto, estar em uma posição traseira permite que o jogador observe mais jogadores agindo e colete dados importantes antes de tomar sua decisão.
* Probabilidade: no *pré-flop*, quando todos os adversários em posições anteriores desistem, uma mão com força intermediária tem seu valor ampliado visto que existem menos chances de um jogador subsequente possuir uma mão superior. Esse fenômeno permite que jogadores nas últimas posições aumentem com cartas fracas apenas com intenção de roubar as cegas, visto que os adversários freqüentemente não terão cartas boas o suficiente para continuar.
* Pressão: em posições traseiras, é possível que um jogador determine o quanto seus adversários terão que pagar para ver a próximas carta. Assim, os oponentes se sentem pressionados e são induzidos a pensar que será custo chegar até o fim da mão.

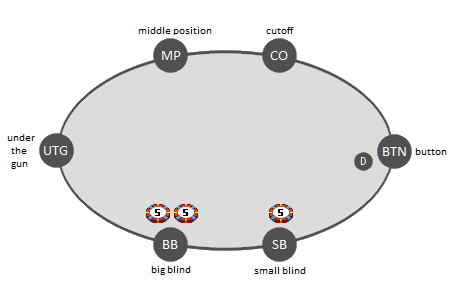


Figura 9 Posições em uma mesa com seis jogadores

Fonte: Elaborado pelo autor

Contrariamente, todos esses fatores são inversos em posições dianteiras. Dessa forma, geralmente é necessário possuir mãos com valor elevado para jogar nessas localizações, procurando reduzir o efeito das desvantagens posicionais. Assim, a quantidade mãos que um jogador consciente entra em jogo é proporcional ao valor de sua posição. Por fim, é necessário salientar que embora as posições sejam fixas, isso não garante que um jogador na última posição (BTN) seja sempre o último a agir. Logicamente, sempre que alguém aumentar uma aposta é preciso que todos os outros tomem suas decisões e, por óbvio, o último a jogar naquela rodada será sempre aquele que estiver à direita do jogador que fez a aposta final. Esse conceito é chamado de posição relativa e, mesmo que deva ser considerado, não afeta nas vantagens de uma posição traseira.

### Análise estatística

Visando explorar as virtudes computacionais, os softwares de análise estatística examinam dados sobre os jogadores e dissecam seu estilo de jogo em números. Esses programas coletam os históricos de mãos enviados pelos servidores de pôquer online e, baseado nos acontecimentos de jogo, enumeram as decisões tomadas por cada jogador em uma base de dados local. A lista a seguir detalha algumas das estatísticas que estes programas proporcionam e serão usadas pelo agente proposto neste trabalho.

* VPIP (*Voluntary Put Money In Pot*) – Porcentagem de tempo que o jogador entra em uma mão voluntariamente. Ou seja, a fração de mãos jogadas.
* PFR (*Pre-flop Raise*) – Percentual de mãos que o jogador aumenta no *pré-flop*.
* AF (*Aggression Factor*) – Fator de agressão, indicando o quão passivo ou agressivo um jogador é. Sua fórmula divide o número de apostas e aumentos pelo número de vezes que o jogador somente pagou (POKERTRACKER, 2010).
* WTSD (*Went to Showdown*) – Parcela de tempo que o jogador, após ver o *flop*, chega até o *showdown*.

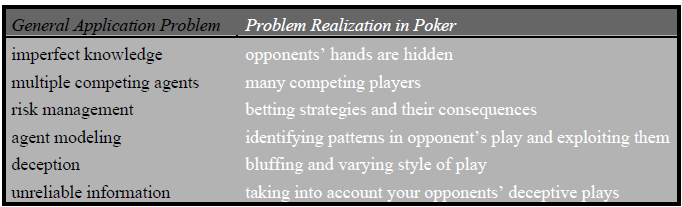
Com base nesses números, um jogador pode identificar com facilidade os méritos de cada jogada. Por exemplo, um adversário com WTSD acima da média costuma pagar mais apostas que o normal. Essa característica pode ser explorada apostando com mãos ligeiramente mais fracas do que a situação recomendaria, visto que, por tendência, o oponente irá pagar com uma vasta amplitude de mãos inferiores. Inversamente, se o adversário possuir um WTSD baixo, as vantagens do blefe são aparentes já que o oponente constantemente desistirá.

Ainda, como os servidores de pôquer online só possibilitam que cada jogador obtenha o histórico das mãos em que esteve presente, algumas abordagens foram propostas para obtê-los mesmo sem participação nos jogos. Conforme Sakai (2005), que apresenta um método para obtenção de históricos apenas observando as mesas, possuir estatísticas sobre um adversário permite que o jogador possa, antes mesmo de enfrentá-lo, conhecer as diferentes nuances do seu estilo de jogo e adaptar-se de acordo.

# AGENTES NO PÔQUER

No passado, os estudos na área de inteligência artificial sofriam duras críticas por se basearem em domínios irrealísticos e utópicos (DAVIDSON, 2002). Jogos como damas e xadrez, com propriedades determinísticas e informações perfeitas, eram alvo de extensivas pesquisas por parte dos estudiosos e posteriormente, o gamão, que possui elementos estocásticos, também foi incluído nesse seleto rol de jogos estudados. Assim, embora a execução desses estudos agregue imenso valor à área, eles raramente podem ser aplicados em situações reais. Algoritmos como *minimax*, *expectimax* e *alpha-beta prune* são algumas das notáveis contribuições advindas dessas pesquisas.

Baseados nesse prognóstico, pioneiros como Darse Billings, Lourdes Peña, Jonathan Schaeffer e Denis Papp focaram seus esforços científicos para outro rumo e introduziram o pôquer como ambiente de testes para a inteligência artificial. A motivação principal para o estudo do pôquer é a semelhança de suas características, possuindo propriedades não determinísticas e informações imperfeitas, com as presentes no mundo real. Isso possibilita que as eventuais descobertas relacionadas ao jogo tenham aplicação em outros problemas. O Quadro 1 apresenta outros aspectos interessantes do ponto de vista de pesquisa encontrados no pôquer.



Quadro 1 Características do pôquer

Fonte: Billings et al, 1998a

Nesse capítulo, será detalhada a progressão histórica dos agentes para pôquer, desde os primeiros protótipos simplistas baseados em regras até a presente situação, com sistemas de busca adaptativa na árvore de jogo, analisando as vantagens e desvantagens de cada arquitetura e destacando os elementos fundamentais que o agente proposto deverá incorporar. Ainda, questões pontuais de desenvolvimento como a modelagem de oponentes e estratégias de apostas são explicadas.

## Propriedades de um agente para pôquer

Para que um agente obtenha sucesso no pôquer, com aptidão para jogar em alto nível, é indispensável que ele incorpore certas características fundamentais de compreendimento do jogo. No capítulo anterior, abordaram-se os conceitos essenciais, do ponto de vista humano, para a correta avaliação das situações presentes em uma partida. Agora, a lista abaixo reconsidera essa perspectiva, enumerando os conceitos de um ângulo prático.

* Adaptabilidade: o sistema deve ser capaz de adaptar-se aos oponentes e as situações, seja por conhecimento histórico ou por observações recentes.
* Modelagem de Oponentes: além de ser um componente elementar para garantir a adaptabilidade, visto que um modelo adequado do adversário permite o ajuste sistemático do agente, a modelagem de oponentes também concede a possibilidade de uma busca com maior precisão pela estratégia de apostas mais lucrativa, através da simulação das ações dos adversários em diferentes situações.
* Imprevisibilidade: o agente deve possuir um estilo de jogo criativo e balanceado, mesclando blefes e jogadas inortodoxas, para impossibilitar a identificação de padrões e induzir os adversários a erros.
* Relativismo: apesar do valor absoluto de cada carta, o sistema obrigatoriamente deve compreender a relatividade intrínseca do jogo. Por exemplo, um par de ases em mesas K♠7♣2♦ e J♥T♥9♥ tem valor completamente distinto.

## Com limite *versus* sem limite

Conceitualmente, as variantes de pôquer com e sem limite apresentam uma única diferença: a flexibilidade no valor das apostas. Na modalidade com limite, os valores são pré-determinados e sujeitos a estrutura das cegas. Já nos jogos sem limite, a única restrição imposta é que o valor de uma aposta (ou o aumento da aposta de um adversário) deve ser maior ou igual ao da última aposta na rodada atual. Quando não houve apostas anteriormente, o valor mínimo é idêntico ao da cega maior. Desse modo, os *pots* tendem a serem maiores nessa variante.

As nuances particulares de cada modalidade tem um grande impacto nas situações de jogo e costumeiramente interferem nas decisões de um jogador, especialmente em fatores como *odds* e consequências de cada erro. Primeiramente, no jogo sem limite, as apostas mantêm-se proporcionais ao tamanho do *pot* e reduzem o número de cenários que um jogador deve pagar baseado somente em *pot odds*. Inversamente, as *implied odds* são ampliadas devido à possibilidade de vitórias significativas nas rodadas posteriores. O contrário acontece na variante com limite, onde o conceito de *pot odds* tem valor significativo, visto que o valor de uma única aposta é geralmente ínfimo em relação ao *pot,* e as *implied odds* são menos significantes já que os ganhos futuros são amplamente limitados ao tamanho das apostas. Por fim, conforme Beattie et al (2007), o jogo sem limite pode punir mais drasticamente as falhas que um jogador comete.

Para a construção de agentes, porém, um aspecto distinto entre os jogos tem determinante importância: a quantidade de estados é muito maior na modalidade sem limite. Isso faz com que seja necessário eliminar alguns estados para o desenvolvimento de agentes competitivos seja computacionalmente realizável. Para melhor demonstrar o significado disso, Gilpin, Sandholm e Sorensen (2008, p. 2) dizem que:

“No pôquer sem limite, um jogador pode apostar qualquer quantia até o valor de fichas que ainda possui. Portanto, em princípio, o espaço de ações de apostas é infinito (já que um jogador poderia apostar um valor fracional de fichas). Mesmo que os jogadores sejam restringidos a quantias integrais de fichas (como acontece na maioria dos cassinos), o número de ações disponíveis é gigante. (O *small blind* possui aproximadamente mil ações disponíveis na hora da primeira ação.) Esse problema não acontece no pôquer com limite e, então, recebeu muita pouca atenção até o momento. [..]” (Tradução nossa)

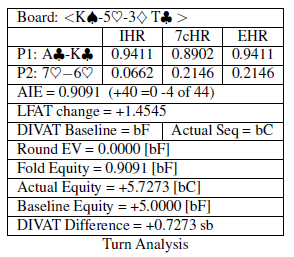
A técnica mais incorporada para tratar esse problema é a de abstração do espaço de apostas (ABOURISK, 2009), que visa limitar a quantidade de ações possíveis aos agentes quando percorre-se a árvore de jogo. Com isso, diminui-se amplamente a complexidade e o espaço de estratégias do jogo. Gilpin, Sandholm e Sorensen (2008) sugere que as únicas ações disponíveis devem ser *check*, *call* e apostas com os seguintes valores: metade do *pot*, total do *pot* e *all-in*. Esses valores devem ser cuidadosamente escolhidos, representando corretamente as quantias e intenções dos adversários.

Finalmente, apesar da preferência dos jogadores pelo Texas Hold’em sem limite, a comunidade acadêmica mantem-se focada na variante com limite. Isso deve-se, em grande parte, a menor complexidade e ao estado de desenvolvimentos dos agentes para a modalidade referida. Em compensação, mesmo com as diferenças citadas entre ambos os jogos, grande parte das técnicas implementadas no pôquer com limite também tem aplicação no jogo sem limite.

## Avaliação de performance

Um dos obstáculos envolvidos na construção de agentes para pôquer reside na avaliação do desempenho dos mesmos. O alto nível de variância do jogo, causado pela presença de elementos estocásticos, não permite que a performance seja estimada considerando apenas os resultados de um jogador (o valor de seus ganhos). Em alguns casos, por exemplo, um participante pode auferir perdas em um curto período de tempo mesmo se sempre optar pela decisão correta, devido à má sorte nas cartas. Ainda, conforme Billings e Kan (2006), é subjetivo determinar qual a melhor jogada em domínios com informações imperfeitas, visto que não se tem conhecimento das cartas do adversário. Portanto, cria-se a necessidade de buscar ferramentas que permitam interpretar a competência de um agente independente dos seus ganhos.

O programa *DIVAT* (*Ignorant Value Assessment Tool)*, descrito em Kan (2007), proporciona um método para a análise pós-jogo da qualidade das decisões tomadas por um jogador. Para tanto, o curso de ação de uma mão é comparado com uma estratégia base adotada pela ferramenta (a estratégia em si é irrelevante e será somente empregada como métrica de desempenho). Desse modo, quando o jogador desvia da estratégia padrão aplicada pelo *DIVAT* e obtém maior lucro, ele é recompensado. Caso contrário, se suas ações resultaram em menores ganhos, ele é punido na análise. O Quadro 2 demonstra uma avaliação para o *turn.*



Quadro 2 Análise DIVAT para P1

Fonte: Adaptado pelo autor (BILLINGS; KAN, 2006)

Nesse exemplo, a ferramenta primeiramente calcula a estratégia base considerando a qualidade da mão de cada jogador (IHR, 7cHR e EHR) em relação às cartas da mesa. Depois, essa estratégia é usada como referência comparativa entre as decisões que realmente ocorreram. No caso, computou-se o curso de ações bF (*bet* e *fold*, isso é, o primeiro jogador aposta e o segundo desiste) enquanto o ocorrido foi bC (*bet* e *call*). Por fim, a equidade de cada estratégia é estimada e equiparada, como ilustram as três últimas linhas do quadro. Assim, como as ações que se sucederam representam maior lucro ao jogador um, ele é recompensado na análise.

Outra técnica freqüentemente aplicada para reduzir o grau de variância do jogo, especialmente em simulações e competições entre agentes, é a permutação de lugares (BILLINGS, 1995). Esse procedimento armazena as informações de uma partida e possibilita que o mesmo jogo seja repetidamente executado, mas com a disposição dos participantes na mesa alterada. Isso é, as cartas distribuídas para cada posição são idênticas e os jogares são transacionados, garantindo que todos recebam as mesmas cartas privadas e comunitárias. Consequentemente, todos os participantes passam pelas mesmas situações e, assim, os efeitos da estocacidade são reduzidos.

## Arquiteturas e desenvolvimento histórico

Muito do conhecimento para o desenvolvimento de agentes para pôquer que os pesquisadores hoje possuem foi adquirido de forma prática, na implementação de suas ideias e teorias. As vantagens e limitações de cada arquitetura, ao contrário do que se pensa, só eram percebidas longo tempo após sua experimentação (BILLINGS, 2006), o que fez com que os aprimoramentos acontecessem de forma gradual.

As seções seguintes, organizadas de forma cronológica, revisam o progresso histórico dos agentes para pôquer, analisando as particularidades de cada arquitetura e ressaltando suas contribuições primordiais. Pretende-se, ao longo dessas seções, não somente narrar o desenvolvimento dos agentes, mas, também, capturar as características fundamentais para a construção dos mesmos. Algumas dessas propriedades, somadas aos conceitos fundamentais de pôquer analisados no capítulo anterior, servirão como base do agente proposto nesse trabalho.

### Sistemas baseado em regras

Após o reconhecimento inicial da adequação do jogo de pôquer para pesquisas de inteligência artificial, os primeiros esforços na criação de um agente para Texas Hold’em resultaram em sistemas simples baseados em regras. Hopgood (2001, p. 19) define um sistema baseado em regras como

“um sistema baseado em conhecimento é um sistema onde a base de conhecimento é representada através de um conjunto, ou conjuntos, de regras. As regras são maneiras elegantes, expressivas, diretas e flexíveis de se se expressar conhecimento.” (Tradução nossa)

Esses sistemas, embora revolucionários à sua época, não consideravam a complexidade inerente do pôquer e as milhares situações distintas que podem ocorrer em uma partida – impossibilitando a criação de regras determinísticas para todas elas. Outro obstáculo freqüentemente presente na criação de sistemas baseados em conhecimento é a necessidade de um especialista humano no domínio do problema. Isso, além de limitar a sabedoria do sistema a do indivíduo, representa complicações na tradução do conhecimento humano para o conhecimento computacional, onde importantes aspectos podem ser mal interpretados.

Por outro lado, o funcionamento de um sistema especialista se assemelha com o processo de racionalização do ser humano, assegurando que sua implementação seja instintiva aos desenvolvedores. Adicionalmente, os agentes baseados em regras realizam pouco processamento e necessitam de mínima quantidade memória. Consequentemente, sua implantação é possível até mesmo em computadores de baixo nível. A subseção seguinte detalha o funcionamento de um agente para Texas Hold’em que possui essa arquitetura.

#### SoarBot

Utilizando uma abordagem totalmente baseada em regras, o SoarBot, construído por Robert I. Follek, integra a tecnologia Soar¹ com o *framework* de testes para agentes de pôquer da Universidade de Alberta. O agente é composto de cento e cinquenta regras básicas, dispostas em cerca de quinhentas linhas de código (FOLLEK, 2001). Interessantemente, o SoarBot executa regras até mesmo para determinar a força de sua mão e as possíveis ameaças na mesa, isso é, se há probabilidade de *flushes*, sequências e outras eventualidades.

De qualquer forma, o SoarBot não possui uma performance notável. Alguns dos fatores limitadores são a inexistência de um especialista no domínio e a incapacidade do agente modelar seus oponentes (FOLLEK, 2001). Essas restrições são especialmente verdadeiras em sistemas baseados em regras, cuja figura de um *expert* é essencial para estruturar as regras. Ainda assim, deve-se notar que o SoarBot foi desenvolvido por apenas um indivíduo, sendo que seus principais adversários eram os agentes da Universidade de Alberta, que possuem um desempenho superior ao de um humano médio.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

¹ http://www.soartech.com

### Sistemas baseados em fórmulas

Os sistemas baseados em fórmulas podem ser considerados uma ramificação dos sistemas baseados em regras. Porém, ao invés de utilizarem regras explícitas para determinar seu curso de ação, idealizam-se fórmulas que avaliam a situação do agente no jogo. Depois, de acordo com os resultados, calcula-se a ação a ser tomada. Suas vantagens e desvantagens são semelhantes as dos sistemas baseados em regras, mas o agente não se torna tão dependente de um especialista no domínio. A seção seguinte analisa um sistema com essa estrutura, cuja base de funcionamento encontra-se em fórmulas para calcular a força da mão do agente.

#### Loki

Desenvolvido pelos pesquisadores do CPRG (*Computer Poker Research Group*) da Universidade de Alberta, no Canadá, o Loki foi o primeiro agente capaz de jogar a nível satisfatório contra oponentes humanos sem conhecimento profundo sobre o jogo. Para mais detalhes sobre seu funcionamento, consulte Peña (1999), Billings et al (1998b) e Billings et al (2001). A arquitetura do agente é baseada em fórmulas, mas considera também outros aspectos do jogo, como a modelagem de oponentes, no seu processo de decisão. A Figura 10 ilustra a arquitetura da primeira versão do Loki, o Loki-1.

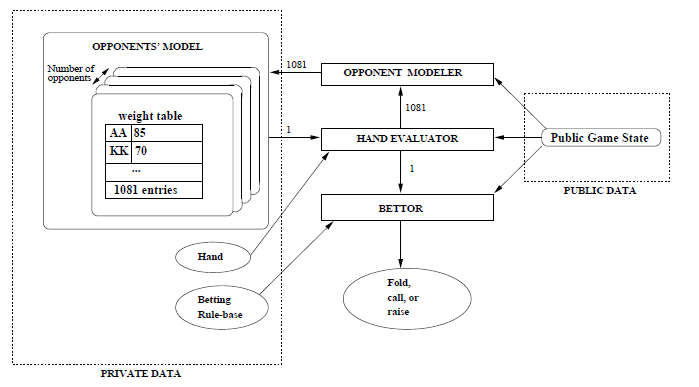


Figura 10 Arquitetura do Loki

Fonte: Peña, 1999

Entre as principais contribuições do Loki para a comunidade acadêmica estão a ideia de avaliação de mãos e a modelagem estatística de oponentes (BILLINGS, 2006). A avaliação de mãos consiste em uma fórmula matemática para determinar a qualidade de uma mão, considerando a força atual e o potencial futuro da mesma. Interessantemente, essa atribuição corretamente não considera o valor estático da mão (por exemplo, dois pares), mas avalia seu desempenho contra todas as outras mãos possíveis. Assim, o valor relativo da mão tem maior relevância.

A Figura 11 ilustra as fórmulas para o cálculo da força (FM) e do potencial, positivo (POTP) e negativo (POTN), de uma mão. Por fim, é demonstrado o cálculo para se chegar à força efetiva dessa mão (FEM), que será usada para determinar a ação do agente nas rodadas *pós-flop*. Essas decisões são tomadas através de um sistema simples de regras que associa um *threshold* de FEM para cada ação (desistir, pedir mesa/pagar, apostar/aumentar).

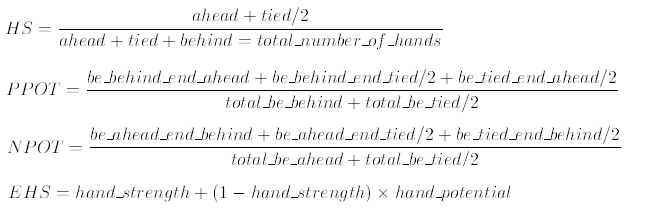


Figura 11 Fórmulas de avaliação de mãos

Fonte: Peña, 1999

Para a modelagem de oponentes, o agente mantem uma tabela com todas as possíveis mãos do adversário e o peso de cada uma delas. Conforme Peña (1999), o peso de cada mão reflete a possibilidade de um oponente específico possuí-la. Primeiramente, os pesos são atribuídos de acordo com a porcentagem de mãos jogadas, avaliando a frequência que o adversário desiste, paga ou aumenta no *pré-flop*. Nas rodadas posteriores, é realizado um processo de repesagem após cada ação tomada, que continuamente atualiza a probabilidade de cada mão em particular. Em Billings et al (1998b), são executados testes para demonstrar a competência das versões do Loki-1 com e sem modelagem de oponentes, resultando em uma larga vantagem para o agente com modelagem.

Ainda assim, o Loki-1 possuía algumas limitações. Seu sistema de tomada de decisões era totalmente determinístico e, ao alimentá-lo com entradas idênticas, as ações resultantes sempre seriam as mesmas. Isso tornava o agente demasiadamente previsível, possibilitando bons jogadores de identificarem padrões e explorá-los ininterruptamente. Além disso, o modelador de oponentes não distinguia corretamente diferentes ações (*calls* e *raises* eram tratados da mesma maneira). Esses fatos impulsionaram a construção da segunda versão do Loki, o Loki-2.

### Simulação

As abordagens anteriores, baseadas em conhecimento específico, alcançaram relativo sucesso e habilitavam a criação de agentes competitivos para o jogo de pôquer – apesar do pouco tempo que o domínio começara a ser estudado. Todavia, essas metodologias possuíam interesse científico limitado, considerando que os sistemas não eram capazes de aprender ou raciocinar por si só (BILLINGS, 2006). A constante necessidade de intervenção humana para solucionar situações críticas no seu funcionamento, aplicando conhecimento especialista sobre o jogo para que o mesmo pudesse interpretar novas situações, constituía um grande entrave na obtenção do desempenho pretendido pelos pesquisadores.

Assim, a terceira proposta de desenvolvimento foi fundamentada na ideia de simulações extensivas. Peña (1999) diz que “uma simulação consiste em jogar uma mão em diversos cenários, do estado atual do jogo até o seu fim, para determinar quanto dinheiro cada decisão irá ganhar ou perder”. Simplificadamente, pode-se dizer que as simulações modelam a árvore completa de jogo para cada uma das várias situações existentes, calculando a média de lucratividade de cada ação. Porém, é inviável computacionalmente percorrer a árvore em todos os diferentes cenários, isso é, todas as possíveis cartas privadas do adversário e futuras cartas comunitárias, sendo que as soluções para esse problema variam em cada implementação.

#### Loki-2 e Poki

A segunda versão do *Loki* e, posteriormente, o *Poki* foram os primeiros agentes de Texas Hold’em com limite da Universidade de Alberta a empregarem o conceito de simulação em suas arquiteturas. O *Poki*, representado na Figura 12, possui três alterações em relação ao *Loki-1*: um simulador, a estrutura de dados *Probability Triple* e uma tabela de mapeamento das frequências de ação dos adversários. Adicionalmente, o *Loki* perdura a estrutura baseada em fórmulas dos seus antecessores, que pode substituir o método de simulação se desejado.

Como é inexequível simular todas as ramificações da árvore de jogo, seus desenvolvedores utilizam a técnica de amostragem seletiva. Essa técnica faz com que a associação de cartas privadas para o adversário nas simulações se mantenha proporcionalmente compatível com a distribuição presente na tabela de pesos de cada oponente (BILLINGS et al, 2001). Ou seja, garantindo que as mãos mais prováveis tenham preferência nas simulações, já que, como aparecerão mais freqüentemente, seus resultados tem maior relevância.

Em cada simulação, é necessário supor as ações posteriores dos adversários para se chegar até a conclusão do jogo. Para isso, o simulador do *Poki* consulta a tabela de frequência de cada oponente e escolhe uma ação randomicamente. Esse procedimento é igualmente realizado para as decisões futuras do próprio agente. Por último, é preciso ressaltar que, para cada amostra, são executadas duas simulações. Uma para determinar o valor esperado da ação *check/call* e outra para calcular a lucratividade do *bet/raise*. O *fold* sempre possuíra EV neutro, visto que não há lucro nem prejuízo futuro.

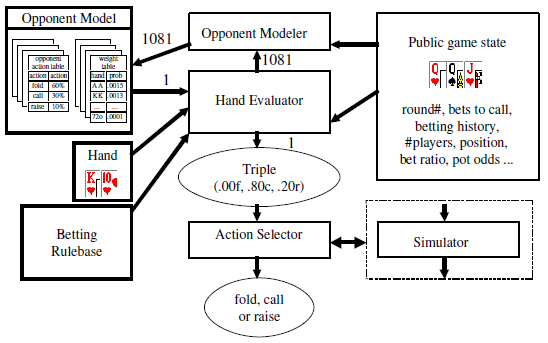


Figura 12 Arquitetura do Poki

Fonte: Billings et al, 2001

### Métodos baseados em teoria dos jogos

É consensualmente aceito pelos cientistas que a teoria dos jogos foi oficialmente formalizada por John Von Neumann e Morgenster no clássico livro *The Theory of Games and Economic Behavior*. Por definição, essa área proporciona um conjunto de ferramentas analíticas úteis na compreensão dos fenômenos existentes no processo de tomada de decisões e na interação entre os participantes desse processo (OSBORNE; RUBINSTEIN, 1994). Sendo assim, a teoria dos jogos é freqüentemente empregada na resolução de problemas de julgamento, muito encontrados em domínios econômicos e em jogos.

Talvez um dos mais importantes conceitos estabelecidos por essa área, e certamente o mais relevante na construção de agentes para pôquer, seja o de Equilíbrio de Nash. Essa ideia, desenvolvida por John Forbes Nash, caracteriza um estado de jogo em equilíbrio quando nenhum dos participantes tem interesse em alterar sua estratégia em um dado momento. Ou seja, a mudança de estratégia de algum dos envolvidos resultaria apenas em prejuízos para o próprio. Mais informações sobre esse conceito são encontradas em Nash (1950) e OSBORNE e RUBINSTEIN (1994).

No pôquer, o Equilíbrio de Nash consiste de uma estratégia mista, isso é, uma distribuição probabilística entre todas as ações possíveis, para cada uma das distintas situações existentes, cujo emprego assegura a obtenção do valor teórico do jogo independente da estratégia adversária (BILLINGS et al, 2003). Simplificadamente, é uma solução ótima para cada nó da árvore de jogo que, quando aplicada, garante ao menos uma expectativa neutra. Essa estratégia é calculada considerando um oponente onisciente que sempre realiza a ação correta. Portanto, não há prejuízos ainda que o adversário jogue de forma perfeita, possibilitando ganhos significativos caso o mesmo desvie da estratégia ótima.

Entretanto, uma árvore de jogo para *Texas Hold’em* com limite e apenas dois jogadores possui aproximadamente estados não terminais (JOHANSON, 2007), ampliando consideravelmente para mais jogadores ou em jogos sem limite de apostas. Dessa forma, ponderando que cada estado necessitará de um perfil estratégico, a aplicabilidade desse método é proibitiva apenas estimando as restrições de memória, podendo ser totalmente impossibilitada ao se analisar as limitações de tempo e processamento. Assim, até recentemente, a busca por soluções ótimas era somente aplicada em problemas de menor escala, como em Koller e Pfeffer (2007).

A solvibilidade do pôquer através de Equilíbrio de Nash com os recursos atuais é, então, dependente de técnicas de abstração para abrandar a complexidade do problema e viabilizá-lo computacionalmente. É importante que a estrutura básica do jogo abstraído mantenha-se semelhante a do jogo completo para que a eficiência das estratégias calculadas não seja afetada negativamente. A lista a seguir caracteriza e esclarece algumas das técnicas de abstração empregadas em outras publicações.

* Abstração de Ação: nos jogos com limite de aposta, se deduz o número de máximo de apostas e aumentos possíveis por rodada, que geralmente é configurado em quatro, para um número inferior que não represente mudanças drásticas no valor esperado de cada jogador (BILLINGS et al, 2003). Já nos jogos sem limite de aposta, também é realizado o procedimento descrito na seção 2.2, que restringe o número de ações disponíveis para cada participante.
* Remoção de Rodadas: desconsidera uma ou mais rodadas de apostas, eliminando a necessidade das computações referentes às mesmas. Quanto mais rodadas são retiradas, menor será a precisão das apurações estratégicas calculadas. Depois, simula-se a conclusão da mão para todas as combinações de cartas restantes nas rodadas eliminadas, atribuindo um modelo pré-determinado de ação (tipicamente nenhuma ou uma aposta de cada jogador) e concedendo a fração esperada do *pot* para cada participante de acordo com sua probabilidade de vitória. Isso é executado para reduzir os efeitos da redução de complexidade do jogo. (BILLINGS et al, 2003).
* *Bucketing*: concilia os estados de jogo de cartas similares, agrupando-as em um único conjunto de equivalência que compartilha o mesmo plano estratégico. Por exemplo, as mãos 7♠6♣ e 7♦6♥, embora diferentes, partilham de atributos suficientemente semelhantes para serem jogadas da mesma maneira. A quantidade de conjuntos distintos, ou *buckets,* determina o grau de abstração a ser obtido. Conforme Johanson (2007), a utilização de um número pequeno de conjuntos permite que as mil trezentas e vinte e seis combinações possíveis de mãos no *pré-flop* sejam reduzidas a um número tratável, como cinco ou dez.

#### PsOpti2

O PsOpti2 é referido como o primeiro agente da Universidade de Alberta para Texas Hold’em com limite capaz de vencer bons jogadores humanos e atuar competitivamente contra um jogador de nível mundial (KAN, 2007). Sua arquitetura baseada em teoria dos jogos obteve um desempenho aclamado e constituiu um grande avanço em relação aos agentes anteriormente produzidos. Para representar simplificadamente a arvore de jogo, as três espécies de abstrações citadas na seção anterior são aplicadas. A Figura 13 ilustra a composição dos modelos abstratos adotados na busca por estratégias ótimas.

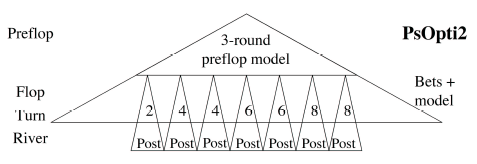


Figura 13 Composição do PsOpti2

Fonte: BILLINGS, 2006

Como não era viável algoritmicamente explorar a árvore completa de jogo, mesmo que abstraída, para encontrar o Equilíbrio de Nash em cada situação, os desenvolvedores do PsOpti2 optaram por subdividir a estrutura básica do jogo em dois modelos simplificados de três rodadas e calcular as soluções previamente, apenas consultando-as em tempo real para determinar a melhor jogada. O primeiro modelo, denominado modelo *pré-flop*, eliminava a última rodada do jogo e era aplicado apenas no *pré-flop*. Segundo Billings et al (2003) excluir as apurações executadas no *river* não influencia significantemente nas estratégias adotadas para a primeira rodada de jogo. Em contraste, no modelo *pós-flop*, o *pré-*flop é ignorado e adota-se um dos sete modelos computados para essa situação, de acordo com o número de apostas no *pot* e qual jogador aumentou por último.

O maior problema dessa abordagem é a desconexão entre os modelos. Cada uma das sete representações *pós-flop* associa uma distribuição probabilística de mãos ao adversário e procura por soluções ótimas com base nessas informações. Porém, é perigoso assumir uma faixa de mãos para o adversário considerando apenas o número de apostas no *pot* e quem cometeu o último *raise*, desprezando conhecimentos contextuais e específicos sobre o oponente*.*

Ademais, os princípios normativos dos métodos baseados em teoria dos jogos são bastante combatíveis já que não buscam maximizar os lucros, mas sim evitar os prejuízos. A suposição hipotética de um adversário onisciente tem caráter impeditivo na admissão de um sistema de modelagem de oponentes e, dessa forma, a exploração deliberada das falhas do adversário não ocorre. Billings et al (2003) apresenta um pensamento condizente ao supracitado, afirmando que não há garantia se o agente será especialmente eficiente contra oponentes ruins ou previsíveis, apenas que o mesmo não irá perder no longo prazo. Assim, mesmo contra péssimos adversário, a margem de lucro pode manter-se comedida.

#### GS1

Andrew Gilpin e Tuomas Sandholm, da Carnegie Mellon University, adotaram uma metodologia semelhante à proposta pelos pesquisadores da Universidade de Alberta na elaboração do agente GS1. Ainda assim, algumas importantes diferenciações se sobressaem, como o algoritmo de abstração automática *GameShrink,* a mínima necessidade de conhecimento específico no desenvolvimento do agente e a aptidão para executar cálculos de equilíbrio em tempo real.

Para produzir uma árvore de jogo reduzida e adequada para a computação aproximada do Equilíbrio de Nash, o *GameShrink* recebe um *threshold* relativo ao nível de abstração a ser calculado. Um *threshold* de zero mantêm o jogo em seu estado original, enquanto um valor infinito agruparia todas as situações em uma só. Portanto, a compreensão profunda sobre o domínio é dispensável para alcançar as abstrações necessárias. Restritivamente, as aproximações são efetuadas apenas para as duas rodadas iniciais, o *flop* e o *turn,* sendo realizadas previamente a execução do agente. Ainda assim, conforme relatam Gilpin e Sandholm (2006), as computações estendem-se por dias e consomem uma quantidade considerável de recursos. Por exemplo, apenas o cálculo da programação linear referente à árvore de jogo reduzida pode alongar-se por sete dias e carecem de dezoito *gigabytes* de memória *RAM.*

Nas rodadas posteriores, o equilíbrio é calculado em tempo real. Primeiramente, utilizando a Regra de Bayes, é encontrada a probabilidade cada mão do adversário baseado no histórico de apostas até o momento. Depois, essas estimativas são aplicadas na resolução do programa linear em um *thread* distinto que pode ser interrompido sempre que necessário (ou seja, no turno do agente). Teoricamente, esse procedimento garante um cálculo de equilíbrio mais preciso para o *turn e river* do que a técnica de *bucketing*, já que considera as cartas exatas que foram distribuídas (JOHANSON, 2007).

A capacidade de cessar as computações do agente em meio aos cálculos de equilíbrio para obter uma decisão imediata, apesar de acrescentar flexibilidade e dinamismo, torna o sistema suscetível a uma falha em particular, o escape da árvore (GILPEN; SANDHOLM, 2006). Supondo que em um determinado momento, quando os cálculos são suspendidos, o agente determine que a jogada ideal seja pedir mesa. Depois, quando o processamento tem seguimento, a programação linear indica que a melhor decisão era, ao contrário, apostar. Consequentemente, no próximo turno, o sistema encontrar-se-á em situação fortuita e que, na sua concepção, inexiste, impossibilitando a estimativa de equilíbrio.

### Busca adaptativa em árvores de jogo com informações imperfeitas

A última abordagem no desenvolvimento de programas para pôquer que será aprofundada nesse trabalho atribui maior relevância a modelagem de oponentes e, concomitantemente, emprega técnicas semelhantes às usadas na resolução de outros jogos estocásticos, como o algoritmo *Expectimax*, objetivando afeiçoá-las para domínios que também contenham informações imperfeitas. Ainda, visando maior aplicabilidade das contribuições resultantes dos seus estudos, os cientistas empenham-se na busca por soluções com pouco ou nenhum conhecimento específico.

Em suma, a tendência atual na construção de agentes reside em sistemas adaptativos e capacitados ao aprendizado. A modelagem de oponentes, cuja importância já havia sido identificada em estudos como Davidson (2002), regressa ao panorama de desenvolvimento como principal instrumento na identificação de padrões e falhas no estilo de jogo dos adversários. Sobre os méritos da MO no pôquer, Billings (2006, p. 123) afirma que

“[..] é uma faceta crítica do bom jogo. Como um jogador possui informaçãos imperfeitas (não sabe as cartas do oponente), qualquer informação que um jogador possa colher do histórico passado de jogadas do adversário permite aprimorar a qualidade das decisões futuras. Uma modelagem de oponentes habilidosa é frequentemente o fator diferencial entre jogadores de nível mundial.” (Tradução nossa)

#### Vexbot

O algoritmo *Miximix*, desenvolvido pelo CPRG e descrito em Billings (2006), alicerce primordial do funcionamento do *Vexbot*, atravessa a árvore de jogo para todas as possíveis futuras cartas comunitárias e, usando um modelo do oponente para prenunciar as decisões adversárias, estima o valor esperado de cada ação disponível ao agente. Ao contrário do que acontece nos métodos baseado em teoria dos jogos, onde teoricamente é necessário percorrer a árvore completa para cada par de cartas privadas do oponente, o mecanismo de busca adaptativa somente requer computações para todas as combinações de cartas comunitárias existentes, a partir do ponto em que a mão se encontra. Dessa forma, é concebível calcular o EV de cada opção em tempo hábil² sem necessitar qualquer abstração de cartas.

Porém, para que o algoritmo desempenhe sua função adequadamente e seus resultados apresentem confiabilidade, é necessário que os nodos de decisão do oponente sejam manipulados corretamente. Ou seja, ao desenhar internamente a árvore de jogo, a probabilidade de cada ação dos adversários deve ser conivente com a realidade. Para exemplificar, supondo que em um determinado nodo existam três ramificações com os resultados esperados de cem, zero e menos cem, respectivamente. Sabendo que o valor esperado do nodo principal é a soma dos valores esperados de todos seus filhos, a lucratividade do mesmo está diretamente relacionada com a probabilidade de alcançar cada ramo.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

² Aproximadamente um segundo, conforme Billings (2006)

Nesse sentido, os pesquisadores desenvolveram uma estrutura de dados para modelagem de oponentes que, além de perfilar os adversários e predizer as ações que serão tomadas com elevado grau de precisão, contem informações sobre a qualidade da mão do participante. Essa ferramenta, batizada de árvore de contexto, armazena as frequências que um jogador empregou certo curso de ação e a respectiva força das mãos (ver Figura 11) que o mesmo possuía nessas situações. Desse modo, para cada nodo de *showdown* da árvore, é mantido um histograma de força de mão, como a árvore na Figura 14.

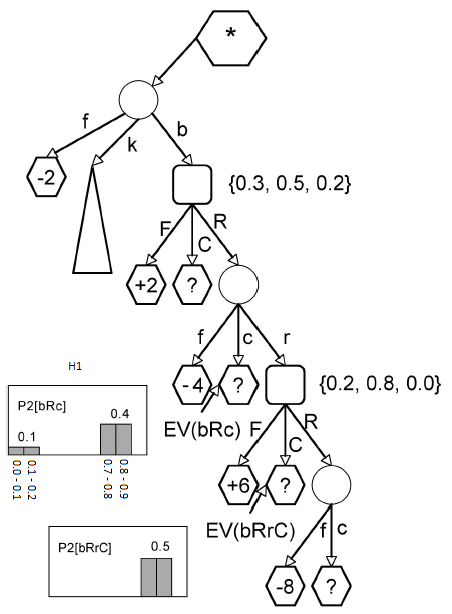


Figura 14 Árvore de contexto

Fonte: Adaptado pelo autor (BILLINGS, 2006)

Na imagem, as letras minúsculas representam as ações do jogador e as maiúsculas são as decisões do oponente. Entre chaves, nos nodos de decisão, as frequências de ação do adversário. Os histogramas nos nodos de *showdown*, ilustrados por retângulos, mantem a periodicidade de cada grupo de mãos. No histograma H1, por exemplo, existe vinte por cento de chance do adversário, após um histórico de apostas bRc, possuir uma mão com força de 0.0 (zero) a 0.2 (zero ponto dois). Assim, essa arquitetura permite que o valor esperado das decisões seja calculado facilmente.

Por outro lado, uma árvore completa de *Texas Hold’em* com limite de apostas engloba seis mil quinhentos e sessenta e um nodos de *showdown* (BILLINGS, 2006). Manter histogramas para todos, além de computacionalmente custoso, necessitaria que cada uma das situações ocorresse um número satisfatório de vezes para que as informações adquirissem confiabilidade. Esse requisito, mesmo em ambientes de simulação, é utópico. Portanto, para que a legitimidade dos histogramas convirja rapidamente, os desenvolvedores usam abstrações de histórico para agrupar contextos semelhantes.

# IMPLEMENTAÇÃO DO AGENTE

O desenvolvimento de um agente eficiente para pôquer é uma tarefa árdua e complexa (PEÑA, 1999) , que demanda não só conhecimento computacional – para elaborar as estruturas de dados que formam o sistema artificial – mas, também, ampla aptidão com as regras e estratégias do Texas Hold’em pôquer. Os capítulos anteriores exploraram conceitos de ambas as categorias e, agora, as informações abstratas investigadas serão aplicadas para concretizar a construção de um agente racional. Como trata-se de um processo longo e difícil, optou-se por segregar esse procedimento em duas etapas.

A primeira descreverá a produção de um agente simplificado, com uma estrutura básica e reduzida, mantendo apenas as propriedades realmente essenciais para que o mesmo possa exercer sua função. Depois, após a conclusão da etapa inicial, a infraestrutura desenvolvida será expandida para proporcionar uma inteligência artificial mais robusta e poderosa. As razões por trás dessa escolha são detalhadas posteriormente, bem como o caráter útil da existência de uma versão ordinária do agente. Ainda, promovendo a maior compreensão dessa proposição, a arquitetura de ambos os agentes é ilustrada em sua respectiva seção. Os elementos chaves, que diferem em cada implementação, também são minuciados e o *pseudocódigo* de algumas das rotinas implantadas é demonstrado quando se julga necessário para clarificar as ideias expostas.

## API Meerkat

Nascida de uma iniciativa para a democratização do desenvolvimento de agentes para pôquer, a API *Meerkat* é uma interface de programação de agentes baseada em Java. Seu principal propósito é definir um protocolo de comunicação entre a plataforma de jogo e os agentes, evitando que os pesquisadores necessitem preocupar-se com os detalhes de implementação da lógica do pôquer em si - um conceito muito importante nas competições entre as diversas inteligências artificiais.

Assim, é possível que um programador desenvolva seu próprio agente, conecte-o a um simulador de pôquer que empregue as regras fixadas pela API *Meerkat*, como o programa Poker Academy, e teste-o contra os agentes disponíveis no software ou publicados por outros estudiosos mundo afora (POKERACADEMY, 2010). A lista seguinte exemplifica as principais classes e interfaces presentes na biblioteca, demonstrando também os objetivos de cada uma.

* Action: essa classe define métodos estáticos para criação de novas ações (*check* ou *raise*, por exemplo) e métodos para classificar o tipo de ação de uma instância dessa classe, como *isBet*() e *getAmount*().
* GameInfo: uma interface com os métodos necessários para identificar as informações públicas de uma mão em execução. Possui métodos que disponibilizam as cartas da mesa, o valor do *pot* e o número de jogadores ativos na mão atual.
* Player: Interface que os agentes devem implementar para comunicarem-se com os simuladores. Contém os métodos *getAction*() e *holeCards*(Card c1, Card c2, int seat), que são chamados, respectivamente, para receber cada ação do agente e para enviar as cartas do mesmo quando uma nova mão se inicia.
* PlayerInfo: semelhante a *GameInfo*, mas para identificar as informações referentes a um determinado jogador. Alguns métodos: *getBankRoll*(), *getAmountInPot*() e *isAllIn*().

Maiores detalhes sobre o propósito de cada classe podem ser encontrados na documentação oficial da API, que ainda contem exemplos de desenvolvimento de um agente básico e repetitivo. É absolutamente indispensável aos pesquisadores que planejam a utilização do *Meerkat* que examinem minuciosamente os conceitos presentes nessa ferramenta, de forma a compreenderem como os dados estão dispostos internamente e quais são as mais eficientes maneiras de se extrair e capturar as informações sobre uma mão em jogo.

## Agente simples

O propósito essencial da criação de uma versão reduzida do agente parte de dois princípios básicos. O primeiro, que solidifica os conceitos de *design* de *software*, é servir de base arquitetural para o desenvolvimento da solução final proposta nesse trabalho. Assim, inicialmente, será executada a construção de uma estrutura escalar, cujos componentes sejam expansíveis. Com isso, é possível que o protótipo inaugural do agente já seja constituído de uma arquitetura robusta e que os níveis de complexidades sejam ascendidos gradualmente, de acordo com o progresso do projeto.

Além disso, como as pesquisas em torno dos agentes para pôquer sem limite de apostas ainda são muito recentes e, conseqüentemente, o número de programas de inteligência artificial disponíveis para essa modalidade é bastante escasso (MCCURLEY, 2009), o agente simples também proporcionará a adição de mais um oponente para os testes de desempenho. Ainda, como a versão preliminar não possuirá alguns dos elementos até então considerados chaves para a performance de um jogador artificial, será possível verificar, através de dados qualitativos e quantitativos, o verdadeiro impacto desses componentes. Então, o modelo básico servirá também como referencial comparativo entre as versões.

### Arquitetura

O agente, denominado ReduxBot devido a sua simplicidade de execução e arquitetura reduzida, incorpora conceitos das duas primeiras gerações de programas racionais para pôquer. Aplicando os princípios de padrões de projeto, são determinadas estratégias para duas situações de jogo: pré-flop e pós-flop. Essa proposição, apesar de não apresentar novas ideias no âmbito do trabalho, auxilia na homologação do agente e tem caráter essencial na assimilação da perspectiva de construção de sistemas inteligentes.

De maneira similar, a complexidade inerente na simulação da racionalidade humana sugere que a introdução ao desenvolvimento desses sistemas deve ser efetivada gradativamente, especialmente quando realizada individualmente (FOLLEK, 2003). Um fator determinante para o sucesso desse processo envolve a capacidade reflexiva e a habilidade analítica durante as etapas de aperfeiçoamento do agente ou de sua arquitetura. A história evolutiva da inteligência artificial, bem como a sucessão progressiva do domínio específico de agentes para pôquer, legitima essa hipótese com um quadro positivo de identificação, decomposição e correção de erros - resultando em avanços inimagináveis. O ReduxBot, portanto, apresenta uma estrutura intencionalmente modesta. Seu propósito existencial é a grande margem de evolução que proporcionará às versões seguintes.

#### Pré-flop

A estratégia de pré-flop, semelhantemente ao agente Loki-I, é composta de um sistema baseado em regras. O conhecimento especialista aplicado para formular as determinações condicionais foi compilado de livros estratégicos sobre o jogo e da experiência própria do autor. Especificamente, foram criadas regras condicionais para estabelecer o conjunto de mãos estático que deve ser jogado em cada situação, além da maneira que o agente da ação que o agente deve inferir. Sklansky (1994) formaliza um método de divisão da mãos iniciais em oito grupos distintos, cada qual reunindo cartas com características equivalentes. Conforme Billings (2006), os rankings de Sklansky mostram uma propensão a pequenos erros lógicos. Assim, o autor examina o resultado de simulações pré-flop contra o sistema de grupos anteriormente exposto e defende sua proposta citando objetivos como a eliminação de conhecimento humano e a adaptabilidade que as simulações fornecem.

Deve-se notar, no entanto, que as observações de Sklansky e os experimentos de Billings concernem à modalidade com limite do Texas Hold’em e máximo de até dez jogadores. Apesar das características de cada mão manterem-se inalteradas, o mesmo não pode ser afirmado com relação à frequência em que as circunstâncias de utilização ocorrem e sobre a própria natureza do jogo. Ou seja, embora o modelo com apostas limitadas e um número maior de competidores beneficie as mãos com propriedades multijogadores (aquelas com potencial para completar jogos altos, como flushs e sequências), o modo sem limites proporciona vitórias mais substanciais. Contrariamente, o jogo ilimitado tem um aspecto penoso elevado e um pequeno erro pode acarretar a perda de todas as fichas do jogador.

Entretanto, a elaboração de uma estratégia pré-flop eficaz depende de um exame minucioso de todas as variáveis envolvidas no processo decisório. Ainda, a mesma deve, em tese, possuir um alto grau de flexibilidade e adaptar suas regras na medida em que as condições de jogo também modificam-se.

#### Pós-flop

Nas rodadas posteriores, uma função de avaliação é aplicada para estimar a qualidade das cartas do agente. Esse método considera a mão atual do jogador, as cartas distribuídas na mesa e o número de oponentes ainda em atividade para determinar o curso de ação a ser tomado. As fórmulas empregadas possuem similaridade com as propostas por Peña (1999) e expostas na Figura 11. Esse método de avaliação simula o jogo até o seu final para todas as combinações de cartas possíveis (do oponente e da mesa) e compara o vencedor em cada situação. Ao término, são obtidos índices quantitativos do número de vitórias, empates e derrotas do jogador, que alimentam as fórmulas de avaliação para determinar a qualidade da mão possuída. A diferença na metodologia implementada consiste em não equiparar a mão atual com o conjunto integral de possibilidades de cartas. Em oposição, são evitadas comparações com mãos com pouca ou nenhuma jogabilidade.

O principal motivo dessa idealização é promover maior distinção nos resultados dos cálculos de qualidade e evitar a inclusão de mãos dissimilares no mesmo grupo de ação. Assim, identificando um limiar comparativo exato onde a classificação das mãos é realista e explicitamente conclusiva, é possível inserir no mesmo conjunto todas as mãos cuja maneira de jogar assemelha-se.

Todavia, estabelecer uma porcentagem rigorosa de mãos habilitadas para esse comparativo requer um alto nível de sensibilidade por parte do pesquisador. A utilização de um valor utópico pode excluir mãos freqüentemente presentes no jogo ou incluir mãos raramente vistas no decorrer de uma partida. Qualquer uma dessas situações influenciaria negativamente na legitimidade da fórmula de avaliação. A tabela abaixo apresenta os valores gerados com diferentes porcentagens comparativas.

Tabela 1 Índices de força

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Força da Mão por Referencial** | | | | |
| **Mão/Mesa** | **30%** | **50%** | **70%** | **100%** |
| K♥Q♠ / A♦6♠K♠ | 0.782 | 0.795 | 0.821 | 0.863 |
| 8♣8♠ / 6♥3♣6♦ | 0.895 | 0.882 | 0.875 | 0.879 |
| Q♠T♣ / J♥T♠2♠ | 0.759 | 0.794 | 0.823 | 0.834 |
| 3♠3♣ / J♣3♥A♦ | 0.982 | 0.989 | 0.992 | 0.994 |
| A♣J♣ / 3♦6♦7♣ | 0.548 | 0.551 | 0.532 | 0.559 |

Com base nos dados expostos, é perceptível a distinção de poder estimada quando aplicados diferentes comparativos. Nas mãos que constituem os extremos opostos dos indicativos, isso é, cartas extremamente fortes ou fracas como as retratadas nas duas últimas linhas da tabela, o diferencial entre as avaliações é ínfimo e não excede o nível de tolerância aceitável. Entretanto, em mãos com força intermediária, há uma grande disparidade nas estimativas dependendo das métricas usadas. Por exemplo, nos dois primeiros casos ilustrados na tabela, mãos com valores dissimilares recebem avaliações praticamente idênticas quando analisadas frente à totalidade das mãos existentes. Em oposição, quando comparadas com um montante de apenas trinta por cento dos jogos, as estimativas refletem com maior exatidão a situação em ocorrência. Nas circunstâncias em questão, o primeiro método calcula um valor semelhante para um *overpair* médio e um par de reis em uma mesa com ás, sendo que o par de oitos se encontra em uma posição indubitavelmente favorável. Já o método proposto identifica corretamente os fatores mencionados.

Ainda assim, quando as métricas em discussão são usadas para constatar o valor absoluto das cartas, é aconselhável empregar o método de comparação completa. Do outro lado, quando aplicadas como guia de decisões de um agente, como nessa abordagem simplificada, é preferível a equiparação com um conjunto seleto e realista de cartas. A explicação teórica consiste no conhecimento comum de que as mãos com valores polarizados são jogadas intuitivamente, enquanto que as intermediárias exigem um tratamento analítico mais aprofundado (MILLER, MEHTA e FLYNN, 2009). Ou seja, o curso de ação é evidente com cartas poderosas ou excepcionalmente fracas, mas pode ser nebuloso em posições marginais onde incorporam-se muitas questões subjetivas. Com isso, é vantajoso que as estimativas desloquem-se adequadamente nessas condições de forma a precisar e comparar as situações em sua equivalência geral.

#### Outras considerações

O gerenciamento de estratégias é executado por uma estrutura de controle denominada *ActionChooser*, que administra e repassa as chamadas de método para as classes corretas. Depois, essas funções usufruem de informações privadas (cartas particulares, *stack* e percepção dos adversários) e públicas (cartas comunitárias, *pot*, ações realizadas e *pot odds*) para julgar a melhor decisão entre as disponíveis ao agente. A Figura 15 expõe os detalhes funcionais dessa organização.

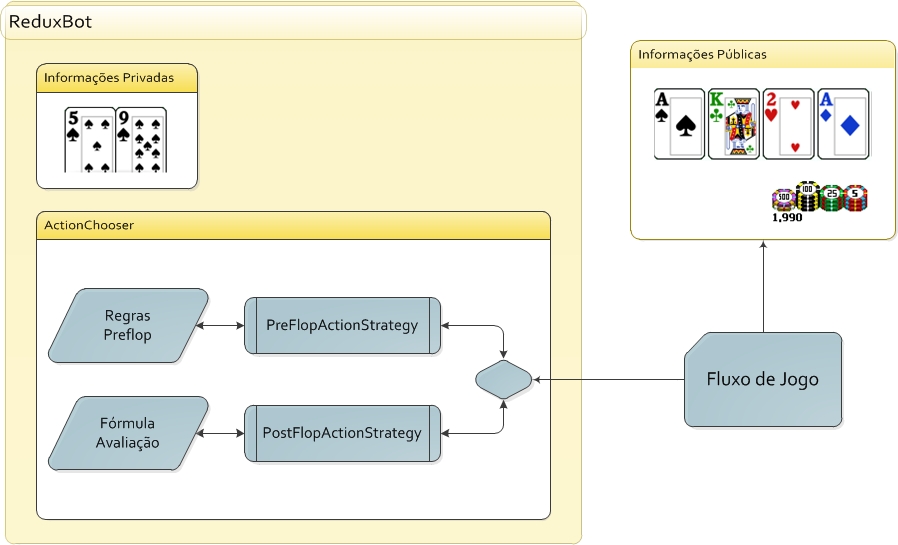


Figura 15 Arquitetura do ReduxBot

Fonte: Elaborado pelo autor

### Principais elementos

Possivelmente, o maior mérito do ReduxBot reside em sua estratégia pré-flop. Embora seja essencialmente estática, ela é baseada em conceitos sólidos do jogo de pôquer (introduzidos por estudiosos de renome) e no conhecimento empírico do Texas Hold’em moderno, que possui uma baixa oferta bibliográfica. O agente, então, aplica ativamente o principio da posição no seu modelo e participa freqüentemente do jogo quando em posições favoráveis, podendo beneficiar-se da abundância de informação e outras vantagens que a mesma proporciona. Por fim, em consonância com o cenário atual do pôquer, o programa foi projetado para sempre preferir a atitude mais agressiva diante de decisões com valor esperado semelhante.

A seleção de mãos, quando aplicada corretamente, pode influenciar diretamente na taxa de ganhos dos agentes baseados em regras – principalmente evitando que eles se envolvam em situações prejudiciais. Por exemplo, um modelo trivial impediria que o agente entrasse no jogo com cartas como K♠J♠ ou A♦T♥ depois que um adversário aumentasse a aposta. Essas mãos, apesar de teoricamente poderosas, são reconhecidas por vencerem pouco quando a frente e perderem quantias substanciais quando derrotadas. Posteriormente, mesmo que formem um par com as cartas comunitárias, elas dificilmente estão em boas condições se o oponente mantem-se apostando. Portanto, uma estratégia que iniba situações como a descrita é certamente valorosa.

Outra contribuição do agente foram as modificações inseridas nas funções de avaliação pós-flop propostas pela Universidade de Alberta (PAPP, 1998 e PEÑA, 1999). A abordagem documentada alterou os modelos comparativos objetivando maior fidelidade nos cálculos de força de mão. Porém, é importante salientar que apesar das mudanças, o método de avaliação é fundamentalmente o mesmo. Ao contrário, a substituição concerne apenas o referencial comparativo empregado e, assim, apenas desloca os índices de força da mão. Embora os resultados dessas alterações mostrem-se positivos, é impossível conjecturar a decorrência exata dos mesmos: se são consequências do cerne da alteração ou do melhor uso dos índices indicativos.

### Limitações

Um aspecto preponderante na arquitetura do ReduxBot é a inexistência de um componente para a modelagem dos adversários. Assim, o agente mantem a mesma estratégia estática contra todos oponentes e é incapaz de identificar e explorar os defeitos presentes em seus respectivos estilos de jogo. A ausência das consequências positivas providenciadas pela modelagem de oponentes, debatidas durante a apresentação do *Vexbot*, impõe obstáculos intransponíveis na aquisição de desempenho qualificado por parte do agente. Ainda, qualquer comportamento minimamente inteligente depende de novas regras e aplicação de conhecimento especialista por parte do autor (FOLLEK, 2003).

Além disso, o processo deliberativo do sistema é determinístico – resultando, invariavelmente, em uma decisão única quando apresentadas situações idênticas. Em razão dessa assimetria decisória, o agente torna-se demasiadamente previsível e explorável (PEÑA, 1999). Ao enfrentar adversários igualmente incapacitados em distinguir padrões, porém, essa característica possui resultados imprevisíveis, podendo auferir lucros exorbitantes ao punir inconscientemente os defeitos no jogo dos outros competidores ou gerar prejuízos na ocorrência do oposto.

Por último, o agente é projetado para não blefar sob quaisquer circunstâncias. Essa preferência arquitetural é fundamentada pelo requisito principal do ReduxBot, a simplicidade. Estruturar a capacidade de blefe, especialmente em um agente determinístico, demandaria a adição de uma grande quantidade de regras específicas e, de maneira implícita, conhecimento especialista. Obviamente, a base de uma estratégia bem-sucedida no pôquer é o balanceamento (MILLER, MEHTA e FLYNN, 2009) e a margem de lucro do programa é negativamente afetada ao abster-se de jogar deceptivamente.

## Agente completo

Apesar da capacidade de jogar competitivamente, o ReduxBot carece de um componente essencial na consecução de um nível de habilidade superior: a adaptabilidade. (JOHANSON, 2007). Fundamentalmente, a modelagem de oponentes serve apenas como uma ferramenta auxiliadora na concepção de um sistema adaptativo. Um agente incapaz de moldar-se as circunstâncias, além de não aproveitar-se dos defeitos dos seus adversários, possibilita que os mesmos explorem os seus eventuais erros e, conseqüentemente, influenciem negativamente nos seus resultados.

Desse modo, a modelagem de oponentes possui caráter de extrema importância na construção de um agente complexo e em conformidade com os propósitos levantados nesse projeto. Para desenvolver o próximo agente, denominado EssentialBot, o ponto central de discussão e alicerce principal da arquitetura conceitualizada será estabelecido na adaptabilidade do sistema. Esse objetivo será consumado através de uma abordagem de modelagem de oponentes baseada em árvores de contexto. Depois, em seção própria, os tópicos e conceitos presentes nesse aspecto constituinte do sistema serão explanados com maior detalhamento.

Finalmente, após a conclusão, a seção de testes projetará experimentos específicos para homologar a capacidade adaptativa do agente, construindo inteligências artificiais reduzidas e claramente previsíveis. Com isso, além dos resultados quantitativos obtidos, os históricos de ação das mãos serão analisados de forma a procurar por indícios da existência do comportamento esperado. Contra oponentes elaborados, no entanto, esse exame torna-se inevidente devido ao desconhecimento do que seria admitido como comportamento explorativo. Por outro lado, ao confrontar agentes previsíveis e determinísticos, é teoricamente concebível determinar uma estratégia ótima.

### Arquitetura

A estrutura arquitetural projetada no desenvolvimento do agente simples, que consolida um mecanismo automático de seleção de estratégias dependendo da etapa em que o jogo se encontra, foi mantida como base para a etapa conclusiva do projeto. A Figura 15, apresentada para ilustrar a disposição do ReduxBot, mantem-se autêntica em seu cerne para caracterizar a composição do agente complexo. Porém, obviamente, as estratégias de aposta foram amplamente estendidas para compactuarem com os objetivos propostos nesse trabalho: a construção de um agente capaz de jogar em nível avançado explorando as tendências alheias. Considerando que a premissa básica dos sistemas baseados em regras ou fórmulas diverge dessa meta, tornou-se evidente a necessidade da construção de estratégias elaboradas que considerem as características dos adversários durante sua deliberação. Os elementos constituintes do sistema são ilustrados na Figura 16.

Devido à extensão da arquitetura desenvolvida, não é possível detalhar e acrescentar todos os dispositivos que compõe o EssentialBot em um diagrama. Sendo assim, algumas das classes sofreram severas abstrações de complexidade. O pacote *bots.util* agrega os módulos compartilhados do agente com seu predecessor, enquanto que as outras partes pertencem unicamente ao mesmo. As estratégias de apostas são pormenorizadas nas próximas seções, bem como os detalhes funcionais que guiam o sistema e sua relação com o diagrama de classes apresentado.

#### Estratégias de aposta

As atividades decisórias do EssentialBot foram desmembradas em dois segmentos únicos referentes às rodadas pré e pós *flop*. A motivação fundamental por trás dessa determinação envolve a inconsistência comumente presente nos modelos estratégicos dos jogadores, que raramente conciliam um estilo de jogo singular durante as duas etapas supracitadas e, de modo oposto, promovem uma perceptível alteração tática após a distribuição das cartas comunitárias na mesa (DAVIDSON, 1999). Logo, torna-se logicamente equivalente segregar as observações e dividir o estilo de jogo do agente também em duas partes, reproduzindo um modelo similar ao dos adversário para capturar com mais eficácia as particularidades dos jogadores. Simultaneamente, a incapacidade computacional de projetar uma árvore de jogo completa desde a rodada inicial, em tempo real, incentiva essa subdivisão.

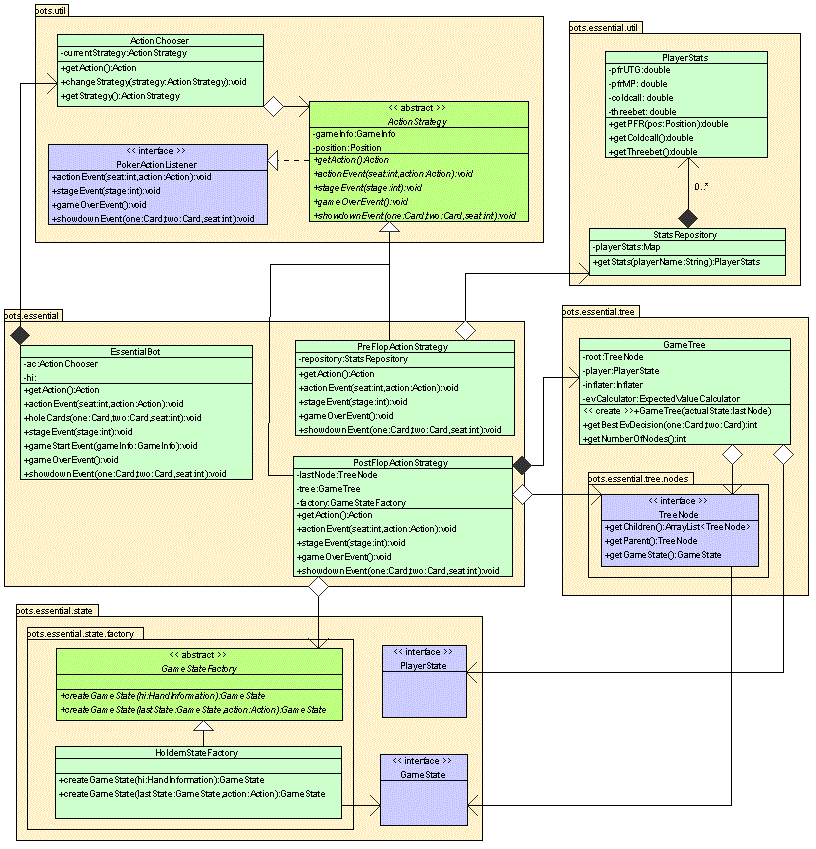


Figura 16 Diagrama de classes abstraído

Fonte: Elaborado pelo autor

Similarmente, uma das propostas secundárias em qualquer projeto de inteligência artificial é evitar a intervenção humana através de conhecimento especialista sempre que possível (BILLINGS, 2006), objetivando na concepção de técnicas deliberativas que possam executar decisões racionais de forma independente. Desse modo, a estratégia do agente para a rodada inicial não consiste de uma tabela estática de ações formuladas com auxílio da bibliografia sobre o domínio pois além de não corroborar com a perspectiva anunciada, essa abordagem é, do ponto de vista do jogo, bastante inflexível.

##### Pré-flop

Em uma explicação mais detalhada, a estratégia pré-flop desenvolvida baseia-se em simulações processadas em tempo real. Sempre que requisitada uma nova decisão por parte do agente, o sistema consulta o estado de jogo atual e avalia a situação em que se encontra. Informações como as ações dos jogadores anteriores e suas respectivas posições, o número de oponentes que ainda devem agir e as estatísticas de todos os participantes da mão são inquiridas e adaptadas antes de serem alimentadas ao componente estratégico, que retorna a opção selecionada. Os dados referentes à situação do jogo são oriundos do próprio ambiente de simulação *Meerkat*, através da interface *GameInfo.* Já as estatísticas de cada jogador são buscadas no banco de dados do programa de análise *PokerTracker,* através da classe *StatsRepository,* que gera informações quantitativas a partir dos históricos das mãos passadas.

O procedimento do componente pré-flop é composto de três etapas. Primeiramente, o agente inspeciona quais participantes anteriores a ele deliberadamente entraram no jogo e suas respectivas estatísticas. Supondo que um adversário aumente a aposta e outro pague-a, o algoritmo buscará os valores de PFR (*Pre-flop Raise)* e CC (*Cold Call)* dos mesmos. Depois, um processo semelhantemente é repetido para os participantes que ainda não agiram. Porém, em vez de consultar as medidas das ações já cometidas, o sistema analisa a probabilidade de cada decisão posterior. Prolongando o exemplo e assumindo que o agente se encontra na posição *Small Blind,* nota-se que há apenas um adversário ainda inativo e são pesquisadas as possibilidades desse participante pagar, aumentar ou sair. Todos os índices adquiridos são então armazenados e alimentam a próxima e última fase de execução.

Após a arrecadação do conjunto de dados influentes para a tomada de decisão, o agente inicia uma série de cálculos com as informações coletadas. Ainda em continuidade com a situação criada anteriormente e admitindo hipoteticamente que a pesquisa indicou um PFR de dezesseis por cento para o primeiro oponente, um CC de quatro por cento ao segundo e uma probabilidade de três por cento para que o próximo adversário continue no jogo, esses valores abasteceriam os próximos passos do procedimento.

Outro fator obstrutivo refere-se à conversão dos percentuais estatísticos dos jogadores em faixas de mãos concretas. Ainda, é preciso que o sistema possua discernimento de que um valor de quatro por cento de PFR e CC representa mãos com valores absolutamente distintos, visto que as cartas possuídas quando paga-se uma aposta e as cartas usadas para aumentar a mesma tendem a pertencer a grupos diferentes. Logo, é apropriada a construção de um quadro referencial de mãos, ordenado com relação à força pré-flop, para que esses índices percentuais possam identificar em qual categoria estão inseridos.

Nesse sentido, conforme proposto pro Sklansky (2006), são executadas simulações *roll-out* para definir o lucro potencial de cada dupla de cartas e, depois que os resultados são estimados e armazenados, o sistema apenas verifica esses dados para ativar o processo de reificação das estatísticas em conjuntos de mãos. Esse procedimento ignora todas as ações nas rodadas conseguintes do jogo e assume que todos os jogadores apenas pagam no *pré-flop*. Então, distribui-se todas as cartas comunitárias na mesa e o vencedor da mão é verificado, atribuindo-o os ganhos auferidos. Depois de uma quantidade satisfatória de repetições, obtém-se o valor esperado da decisão e a frequência de vitória das cartas testadas. Deve-se ressaltar que o procedimento *roll-out* pode ser aplicado em comparações entre duas faixas de mãos, entre duas mão únicas ou entre uma mão em particular e um conjunto extenso de mãos, como aplicado por Sklansky em seus experimentos.

Então, o mecanismo decisório pré-flop considera qual ação adversária foi cometida e avalia sua provável faixa de mãos. Em circunstâncias de *raise* pré-flop, de acordo com a natureza da jogada, intenta-se selecionar o conjunto com as mãos mais fortes que representem a frequência de aumentos do adversário, conforme mostra a Figura 17. Para isso, seleciona-se o percentual exato de mãos referentes ao PFR do oponente. Por exemplo, a faixa de mãos de um adversário com pré-flop *raise* de dez por cento constituir-se-ia das cento e trinta e duas³ mãos mais fortes presentes no quadro referencial. Contudo, em situações cujos participantes apenas pagam um *raise*, essa transformação exige cuidados especiais pois os fatores que permeiam o processo deliberativo são muito mais nebulosos que na condição anterior. Especificamente, os princípios que guiam as decisões de *call* estão freqüentemente associados a elementos condicionais, como a quantidade de jogadores que já entraram no jogo ou características particulares de um respectivo jogador.

Nesse caso, portanto, existem duas opções viáveis para se estabelecer o conjunto de mãos prováveis do oponente. A primeira abordagem monitora todas as ações de *call* realizadas no pré-flop e, caso o jogador alcance o *showdown*, armazena a mão possuída em uma estrutura de dados. Assim, cria-se um meio fidedigno para consulta de *range* baseado no histórico passado do participante*.* Todavia, a autenticidade dessas informações só é garantida após uma vasta amostragem e, por tratar-se de um jogo parcialmente observável, não há nenhuma evidência de que, de fato, as mãos chegarão até a etapa final do jogo, quando os jogadores revelam suas cartas privadas*.* Como consequência, o agente pode projetar uma estratégia desbalanceada.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

³ Dez porcento das C(52,2) = 1326 mãos possíveis

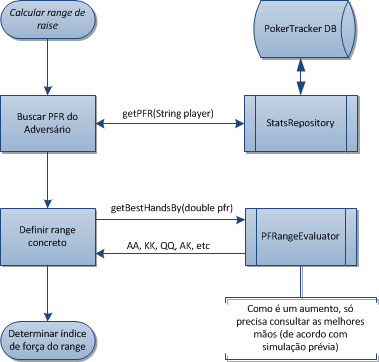


Figura 17 Estimativa de *range* para aumentos

Fonte: Elaborado pelo autor

Já no método implementado no EssentialBot, são identificadas as propriedades que constituem as mãos comumente presentes em situações de *call pre-flop* e atribuídas margens percentuais, inicialmente uniformes, para cada categoria de mãos. Essa distribuição frequencial é atualizada sempre que uma mão chega ao *showdown*. Depois, quando pretende-se recriar a *range* do adversário, o sistema procura no quadro referencial *roll*-*out* pelas melhores mãos que relacionam-se com as tendências adversárias. Exemplificando, se um adversário caracteriza-se por pagar no pré-flop com apenas pequenos pares (uma das categorias de mão), o algoritmo buscará pelas mãos mais poderosas que pertençam a esse grupo segregativo e correspondam ao percentual de *cold call* pré-flop. Como na abordagem anterior, a insuficiência de amostragem também pode reduzir a precisão das conjecturações inferidas, mas, nessa situação, é possível pré-alimentar os sistemas com um padrão probabilístico confiável – seja baseado em históricos passados dos oponentes ou do próprio jogador.

A necessidade de incluir os oponentes que ainda não agiram no processo de decisão impõe um obstáculo adicional na concepção das técnicas supracitadas, visto que os prognósticos de *range* dependem exclusivamente do tipo de ação cometida. Para ultrapassar essa barreira, o algoritmo combina as duas perspectivas anteriores, examinando as chances do adversário pagar e aumentar para determinar um conjunto válido de mãos. Supondo que, na situação exposta na Figura 18, as estatísticas indiquem um percentual de *call* e *raise* de cinco e três por cento, respectivamente, para o último adversário que não agiu (*BB)*, o algoritmo empregaria o procedimento referido anteriormente para cada uma dessas circunstâncias e multiplicaria pela probabilidade de ocorrência para construir uma *range* idônea.

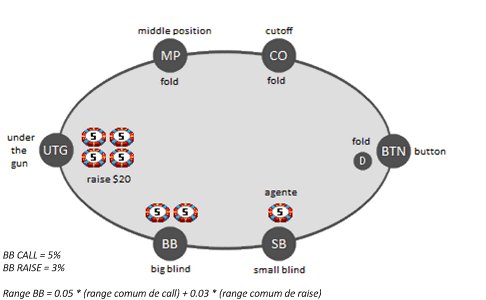


Figura 18 Definindo o *range* de um oponente que ainda não agiu

Fonte: Elaborado pelo autor

Finalmente, com as prováveis faixas de mãos dos jogadores decretadas, o algoritmo pode estimar a média de força de cada *range* e determinar a ação com maior retorno. Desse modo, com base em modelos matemáticos definidos pelo autor, o componente estratégico determina sua ação comparando a força de sua mão com a média dos adversários. Observa-se que, embora contraditório, não deve-se obrigatoriamente aumentar se a força da mão do agente é superior a média adversária. Ao contrário, a *range* que concerne nessas circunstâncias é aquela que continuaria após o *raise*, e não a que o oponente possui no momento. De modo semelhante, para aplicar blefes, considera-se, além da média de força, a chance do adversário desistir (uma das estatísticas mantidas pelo *PokerTracker).* Ainda assim, como o principal mérito do agente é sua habilidade pós-flop, é preferível abdicar-se de decisões que não sejam evidentes nessa etapa inaugural do jogo para que o mesmo possa auferir de suas vantagens posteriormente.

##### Pós-flop

O maior desafio na concepção de um sistema inteligente para pôquer consiste no desenvolvimento de estratégias para as rodadas posteriores do jogo. Além de todas as propriedades que incentivam o estudo do domínio, como a imperfeição das informações, a variância intrínseca e a existência de múltiplos competidores, a perspectiva pós-flop agrega, ainda, uma explosão de estados na árvore de jogo (BILLINGS, 2006). Essa quantidade abundante de circunstâncias representa um panorama de complexidade que vai muito além da compreensão humana. Não coincidentemente, os sistemas baseados em regras, que constituem uma abordagem extremamente condicional, defrontam problemas até mesmo na obtenção de um nível de jogo regular.

Agora, com o irrefutável amadurecimento das pesquisas referentes à área e a transição natural para os sistemas explorativos, a compreensão dos adversários torna-se o alicerce fundamental no desenvolvimento de um agente (DAVIDSON, 2002). Com base nisso, a estratégia pós-flop proposta concilia um mecanismo complexo de modelagem de oponentes e reverte a situacionalidade inerente ao pôquer em prol do sistema – mais especificamente a composição das cartas comunitárias e sua influência na estratégia adversária. Desse modo, sempre que necessária uma nova decisão por parte do agente, as informações de jogo são repassadas ao componente de modelagem de oponentes, que executa simulações considerando o histórico passado dos adversários e os dados da mão em questão. Os detalhes funcionais desse mecanismo, constituído de uma árvore de contexto, são minuciados em seção posterior.

##### Tamanho das apostas

Uma estratégia bem-sucedida deve, adicionalmente, considerar o tamanho de cada uma das apostas inferidas para maximizar os lucros obtidos. Esse aspecto do jogo tem papel análogo a uma negociação financeira e engloba fatores incisivos como a probabilidade do adversário pagar ou desistir frente a uma determinada aposta e sua percepção dos oponentes. Ainda, há a presença do caráter psicológico envolvido nos valores, como a hipótese de que apostas baixas raramente são atribuídas a blefes já que possuem maior propensão de serem pagas. Como o foco recente das pesquisas está voltado aos jogos com limite fixo ao valor das mesmas, esse elemento preponderante do jogo não recebeu notável atenção e mantem uma base de teorias bastante escassa, resultando em uma boa margem para pesquisa.

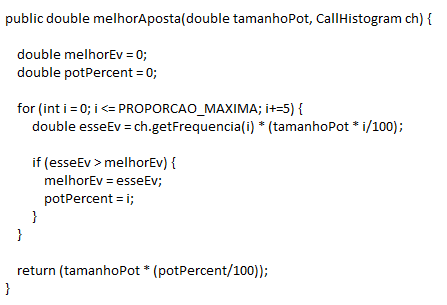


Figura 19 Procedimento melhor aposta

Fonte: Elaborado pelo autor

Para aproveitar-se disso, o agente armazena quatro distintas tabelas de frequência, referentes a cada uma das rodadas do jogo, contendo a distribuição probabilística de cada decisão dos oponentes ao defrontarem-se com uma aposta e o valor proporcional da aposta em relação ao *pot*. Esse histograma é consultado diante de cada decisão e, de acordo com as intenções do agente, uma função matemática específica é aplicada. Quando executa-se um blefe, a quantia com menor chance de *call* deve ser buscada. Em circunstâncias opostas, quando deseja-se que o oponente continue no jogo, o procedimento deve descobrir o valor cuja quantia multiplicada pela probabilidade de ser pago seja a mais elevada, conforme a fórmula da Figura 19.

#### Modelagem de oponentes

Em contraposição aos jogos com informações perfeitas, cujo conhecimento sobre as tendências adversárias não exerce grande influência no processo de decisão de um jogador, a modelagem de oponentes tem um grau de importância notável em domínios imperfeitos. No clássico jogo de xadrez, a acessibilidade do tabuleiro permite enumerar todos estados de jogo futuros e torna evidente a irrelevância de informações específicas sobre o comportamento alheio. Ao contrário, no pôquer, o processo de conjecturação das situações subsequentes depende exclusivamente de suposições relacionadas aos adversários. Essa distinção característica entre as categorias de jogos obrigou que um novo campo de pesquisa fosse formado, estudando o que agora é conhecido como modelagem de oponentes.

Segundo Davidson (1999), a modelagem de oponentes pode ser usada de pelo menos duas maneiras diferentes no pôquer. Primeiro, como um método geral para deduzir a força da mão de um oponente baseado em seu comportamento anterior. Segundo, para prever uma ação futura diante de uma situação específica. Por exemplo, um sistema de modelagem poderia predizer o quão propenso a blefar um jogador está ou que tipo de mãos é possível esperar que o mesmo possua em determinadas circunstâncias. A lista abaixo exemplifica as metodologias comumente empregadas para execução dessa tarefa de acordo com Davidson (2002).

* Sistemas especialistas: utilizam as estratégias de apostas do próprio jogador para determinar as ações dos adversários. Não é um método eficiente, mas fornece uma base comparativa.
* Estatística: armazena a distribuição probabilística de cada ação do oponente em uma tabela e admite que o mesmo persistirá com o mesmo comportamento nas decisões futuras.
* Rede neural: modelo matemático com auto-aprendizagem capaz de identificar as ações adversárias ao ser alimentado com informações relevantes. Em estudo preliminar, planejava-se utilizar essa abordagem no agente completo, mas uma estrutura de dados mais eficaz foi preterida.
* Árvores de decisão: estruturas intuitivas freqüentemente aplicadas como ferramentas auxiliadoras no processo de decisão. Cada nodo apresenta um questionamento sobre o estado de jogo e a ramificação correta é aprofundada até que se chegue ao nodo de decisão do adversário, que conterá as frequências de cada ação.
* Redes Bayesianas: observa as estratégias do adversário sob determinadas situações e associa esses valores a um modelo probabilístico. Depois, o sistema consulta esse modelo para determinar a melhor resposta à decisão oposta (SOUTHEY et al, 2005).
* Árvores de contexto: descritas na apresentação do Vexbot, são árvores de jogo que conciliam a probabilidade de cada nodo com histogramas de força de mão observados empiricamente (BILLINGS, 2006). Essa abordagem será estendida e aplicada no agente complexo.

Uma peculiaridade presente em todas as abordagens recentes para execução da modelagem de oponentes é que apesar de consultarem informações pontuais sobre o estado de jogo, nenhuma concede a devida importância à textura das cartas comunitárias. Como textura entende-se a coordenação entre as cartas e o número de mãos potenciais que se correlacionam com a mesa. Por exemplo, uma mesa composta das cartas K♠7♦2♥ possui um baixo nível de coordenação devido a inexistência de quedas (é impossível completar uma sequência ou *flush* na próxima rodada) e apostas exageradas nessas circunstâncias restringem a faixa de mãos do jogador para cartas extremamente fortes ou fracas, como trincas ou blefes. Conclusivamente, a textura da mesa permite segregar a situacionalidade do jogo e agrupar circunstâncias similares em um mesmo espaço de análise.

Não obstante disso, o agente completo agregará um algoritmo para o avaliar a textura das cartas comunitárias entre cinco categorias, ordenadas crescentemente em grau de coordenação e esses índices serão usados, em conjunção com outros dados, para alimentar o componente para modelagem de oponentes que constituí o sistema artificial. A composição desse módulo é explicada na subseção conseguinte, bem como a maneira que aplica os índices de textura na sua tomada de decisão.

##### Árvore de contexto

Basicamente, uma árvore de jogo é um grafo direcionado cujos nodos representam estados, como a disposição das peças no xadrez, e os vértices retratam as jogadas disponíveis em uma respectiva situação. Conforme pormenorizado durante a apresentação do Vexbot, as árvores de contexto são extensões das árvores de jogo tradicionais, agregando informações pertinentes ao contexto de jogo em cada nodo e, conseqüentemente, fornecendo meios para que se estime o valor esperado de cada jogada (BILLINGS, 2006). Para compreender esse conceito corretamente, é necessário, primeiramente, incorporar como os nodos de *showdown* são retratados em uma árvore de pôquer.

Na árvore simplificada, exposta na figura a seguir, onde apenas uma ramificação é aprofundada, estão presentes todas as informações necessárias para a avaliação do valor esperado de cada jogada. Nota-se que, como a mão do oponente é superior, todos os nodos terminais com *call* resultam em prejuízos ao agente, assim como as situações que o mesmo desiste da mão depois de já ter contribuído com dinheiro ao *pot*. Contrariamente, as únicas circunstâncias que o agente aufere lucro são decorrentes de desistências do adversário. No entanto, mesmo com o resultado de cada nodo exposto, a árvore é meramente elucidativa e não fornece os meios para que se delibere a melhor decisão por parte do agente, visto que essa discriminação depende exclusivamente da probabilidade de alcançar cada ramificação.

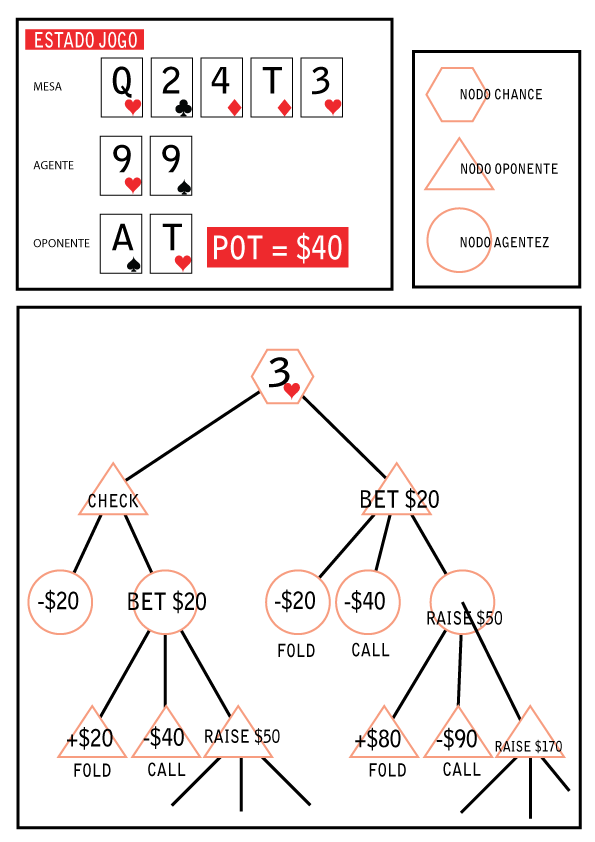


Figura 20 Árvore simplificada

Fonte: Elaborado pelo autor

A existência desses critérios subjetivos, que conduzem o processo decisório, obriga que qualquer sistema baseado em árvore armazene dados referentes à probabilidade de se atingir cada nodo da estrutura para uma avaliação de ação consistente. Para tanto, as árvores de contexto acumulam a quantidade de vezes que um nodo foi alcançado e, desse modo, podem obter um índice de frequência das ações adversárias. Ou seja, a probabilidade de um determinado oponente executar uma ação é dada pela divisão entre o número de vezes que esse nodo foi alcançado e a soma do número de vezes que esse nodo e seus irmãos foram atingidos. Em adição a esses valores, as árvores de contexto também guardam um histograma com a frequência de força das mãos que o oponente possuía ao atingir cada nodo. Essas estruturas de dados mantem, mesmo que de maneira incompreensível humanamente, o *range* dos adversários em cada situação. Portanto, é possível estimar, também, a probabilidade de vitória do agente e, conseqüentemente, o valor esperado dos nodos terminais. A Figura 21 ilustra esses conceitos com uma árvore mais complexa, cujas informações privadas do adversário não estão disponíveis, e os cálculos são realizados com base em histogramas hipotéticos sobre o mesmo.

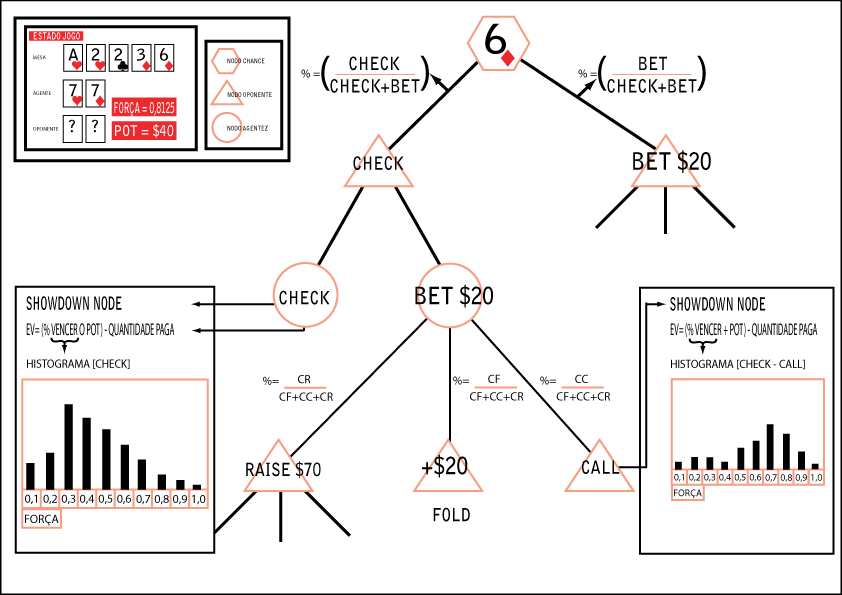


Figura 21 Árvore com histogramas

Fonte: Elaborado pelo autor

Uma distinção fundamental entre as árvores de contexto sugeridas por Billings e aquelas implementadas no desenvolvimento do EssentialBot tangem a modalidade de jogo que são empregadas. Enquanto a primeira é aplicada no Texas Hold’em com limite, o agente aqui proposto frequenta ambientes sem limites de aposta. Obviamente, a transição desse conceito para um novo domínio requer que algumas considerações sejam esclarecidas. O contexto limitado impõe que cada nodo da árvore possua um número pré-definido de filhos (referentes às ações de *fold, check/call* e *bet/raise*). Em contraposição, o paradigma ilimitado caracteriza-se pelas inúmeras possibilidades de valor de aposta. Consequentemente, cada nodo contém uma quantidade abundante de filhos.

Entretanto, como as árvores são construídas em tempo real, torna-se impraticável reproduzi-las fielmente devido a sua grande extensão. Logo, visando à abstração da árvore sem perda excessiva de autenticidade, as opções de *raise* devem ser restringidas para um número apropriado. Mantendo o número de valores possíveis de aposta reduzido, parte-se do princípio que todas as ações dessa natureza são logicamente equivalentes e que a identificação de uma tendência de comportamento referente ao valor apostado é um processo muito custoso. O EssentialBot, em conformidade com essa afirmação, estende apenas um nó de *raise* com valor idêntico ao *pot*. Quando há permissão de mais de uma opção de aposta, é fundamental que hajam ferramentas para determinar a probabilidade de cada valor de *raise*. Caso contrário, os cálculos de EV terão sua credibilidade afetada.

Em referência ao diagrama de classes apresentado na Figura 16, a interface responsável pela expansão hipotética das árvores é denominada *Inflater*, e compõe a classe *GameTree.* Empregando o padrão *Observer,* a estratégia pré-flop mantem uma árvore de jogo atualizada com as ações já realizadas pelos jogadores, onde cada nova ação gera um *TreeNode*. Assim, quando necessita percorrer as possibilidades futuras de jogo, uma implementação de *Inflater* simplesmente recebe o último nodo de jogo, onde pode consultar todas as informações da mão (cada nodo possui uma implementação da interface *GameState*) e gerar uma árvore de jogo completa. Por fim, para estimar o valor esperado de cada jogada, uma classe especializada – *ExpectedValueCalculator –* transgride a árvore e efetua os cômputos inquirindo os histogramas de força de mão.

Segundo Billings (2006), são necessárias abstrações também na maneira em que os histogramas de mão são armazenados. Como cada sequência de ação acontece com frequência relativamente rara, seriam necessárias milhares de amostras até que a veracidade das conjecturações sobre os oponentes seja atestada quando usa-se o armazenamento baseado em sequência. Portanto, os histogramas devem ser associados a outros atributos senão a sequência concreta de ações executadas. Billings sugere abstrações simplificadas, como o número total de ações de *raise* ou aproximações dovalor final do *pot*, objetivando uma rápida convergência.

A composição de abstração do EssentialBot é configurada através de um mecanismo concebido, denominado de pontos de ação. Com ele, cada tipo de ação é associada a um número de pontos que indicam, genericamente, sua força. Um modelo simplificado poderia atribuir três pontos para apostas e aumentos, um ponto para ações de *call* e nenhum ponto para as decisões restantes. Ao fim da mão, a sequência de ações executadas é examinada e o algoritmo calcula o nível de pontos auferidos, atualizando o histograma referente a esse valor. A principal vantagem desse método é que a quantidade de pontuações distintas é limitada e, portanto, a porção de histogramas mantidos para cada jogador também. No entanto, deve-se ponderar atentamente sobre o número de pontos concedidos a cada ação para que as circunstâncias reflitam com legitimidade as encontradas no ambiente de jogo. Por exemplo, no modelo proposto, as ações de *bet* e *raise* são imputadas do mesmo número de pontos, mas, geralmente, o conjunto de mãos possuídas nesses dois casos pode não ser idêntico. Ainda, é recomendável um intervalo maior entre as ações agressivas e passivas, mesmo que essa determinação resulte em uma quantia minimamente superior de situações existentes.

Outra contribuição proposta por esse trabalho às árvores de contexto originais envolve a manutenção simultânea de múltiplos histogramas informativos para cada circunstância de jogo, de acordo com o índice de textura das cartas comunitárias. Desse modo, durante o processo de construção da árvore, em tempo real, os dados consultados seriam concernentes à situação da mesa atual. A motivação por trás dessa proposição consiste em refinar os histogramas de força de mão presentes nos nodos terminais para que representem com maior precisão e autenticidade a distribuição das cartas privadas dos adversários. Consequentemente, os cálculos de valor esperado adquirem exatidão superior. Acredita-se que, dentro de qualquer domínio, os valores que norteiam as decisões dos agentes presentes estão vertiginosamente associados às condições em vigor no ambiente estudado. Portanto, é logicamente compreensível segregar essas circunstâncias e as informações provenientes das mesmas.

Para agentes de pôquer, onde uma árvore pode tranquilamente hospedar centenas de milhões de nodos, é imperativo que as abstrações empregadas reduzam a complexidade mas não removam completamente a situacionalidade do jogo, visto que a mesma é um dos elementos que orientam o comportamento dos competidores – por vezes até mais contundentemente que valores absolutos (como as próprias cartas privadas). Logo, o êxito nesse âmbito depende exclusivamente dos algoritmos de abstração de ação e refinamento de contexto que objetivam identificar computacionalmente a inter-relação de circunstâncias que acompanham o ambiente sem generalizá-lo excessivamente.

Após a execução de um número vultoso de mãos de teste, os históricos de ação dos jogos serão analisados, bem como as estruturas de histograma oriundas desses testes, com pretensões de confirmar as hipóteses supracitadas. Sendo assim, examinando esses dados, será possível evidenciar se o estilo de jogo e as tendências dos adversários alteram-se devidamente na medida em que o ambiente modifica-se.

### Principais elementos

Como pretendido desde o princípio do desenvolvimento do EssentialBot, a concepção de um sistema artificial que não logra de conhecimento especialista é um dos destaques desse projeto. Essa característica permite que todos os conceitos apresentados durante a apresentação do agente possam ser adaptados para outros campos, desde que sejam identificados os elementos que servem como guia decisório das entidades inseridas nesse ambiente. No pôquer, uma propriedade que freqüentemente orienta a mentalidade dos jogadores é a coordenação das cartas comunitárias (nível de textura). Logo, o desenvolvimento de algoritmos para classificar taxonomicamente a mesa, de forma automática, é benéfica para a performance do agente.

Além disso, ambos os componentes estratégicos possuem idéias inovadoras. Primeiro, durante a rodada pré-flop, o sistema decreta o seu curso de ação dinamicamente e de acordo com as propriedades particulares dos seus adversários. A integração com os programas estatísticos concede um grau elevado de autonomia ao agente, que pode consultar informações específicas sobre seus adversários e moldar seu estilo de jogo convenientemente. Em adição, o modelo pós-flop mantem dados sobre a oposição, adquiridos empiricamente, que permitem o cálculo preciso dos valores esperados de cada decisão. A segregação da modelagem de oponentes de acordo com a textura da mesa mostrou-se consideravelmente vantajosa e é um exemplo de como a observação minuciosa do domínio pode ser positivo no desenvolvimento de sistemas inteligentes.

Em retrospecto, o maior desafio presente no desenvolvimento do EssentialBot consiste em conciliar os conceitos de índice de textura com os princípios existenciais das árvores de jogo. Além de buscar por um nível de abstração que não generalize demasiadamente o domínio, tornando situações distintas em semelhantes para a árvore, é preciso manter histogramas de informação para cada circunstância repetidas vezes, proporcionalmente ao número de texturas existentes. Esse objetivo, de atingir a convergência rapidamente e, ainda assim, diferenciar os contextos eficientemente, mesmo que conflitante, pode ser atingido através de uma meticulosa análise dos fatores preponderantes encontrados no ambiente de execução. No agente construído, esses elementos são encontrados e automaticamente caracterizados com os algoritmos de identificação de textura e de abstração de ação.

### Limitações

Uma das maiores restrições que afligem os sistemas adaptativos refere-se ao número de amostras necessárias para que se atinja um índice de precisão confiável nas hipóteses sobre os adversários. Desse modo, ambos os componentes estratégicos do EssentialBot necessitam de um número considerável de mãos até sua convergência. Para abreviar esse empecilho, os adversários são perfilados de acordo com suas estatísticas e, na existência de um jogador cuja amostragem não alcançou um limiar pré-estabelecido, o sistema consulta um modelo com índices quantitativos similares aos possuídos por esse jogador. Assim, embora a margem de erro do valor esperado das jogadas não seja autêntica, já é possível ter alguma compreensão do estilo de jogo do oponente. De modo oposto, quando opta-se por não determinar um número de mãos mínimo antes de usar essas observações como base de cálculo, o agente pode apresentar um jogo muito desbalanceado nos momentos iniciais.

Em adição, apesar dos componentes pré e pós *flop* possuírem um ótimo desempenho, não há comunicação entre ambos e, assim, eles não atuam complementarmente. Essa interindependência resulta na inacessibilidade de algumas informações que poderiam influenciar o processo decisório do agente em situações de risco. Um exemplo corriqueiro seria referente às circunstâncias onde os competidores exercem *raises* no pré-flop e, quando as cartas comunitárias são distribuídas, persistem com jogadas agressivas. Assim, ao verificar os histogramas adversários, o sistema ignora as apostas cometidas na primeira rodada e apenas considera as sequências de ação ocorridas após o *flop*. Uma correção intuitiva a esse comportamento consiste em acrescentar as decisões iniciais nos históricos de sequência. Contudo, mesmo que esse procedimento remova a desconexão entre os componentes, um número considerável de novas sequências de ação são geradas, requerendo ainda mais amostras até que possam ser inferidas suposições precisas nos cálculos de valor esperado executados.

Por último, embora não classifique-se como limitação do sistema, uma observação sobre o modo em que as mãos são avaliadas no pré-flop deve ser ressaltada. Como as simulações *roll-out* assumem que todos os jogadores participam da mão e não executam qualquer ação, simplesmente distribuindo todas as cartas comunitárias e verificando o vencedor, apenas a frequência de vitória de cada dupla de cartas é estimada. Consequentemente, os resultados provenientes desse processo ignoram o potencial de lucro das mãos e prejudicam cartas com propriedades de *implied odds* (que não vencem freqüentemente mas costumam auferir grandes lucros quando o fazem). Assim, o conjunto possível de mãos dos adversários, construído pelo sistema, possuíra uma leve propensão às cartas com características opostas às citadas.

# TESTES E RESULTADOS

Nesse capítulo, serão executados os testes de desempenho referentes aos agentes construídos anteriormente. Após a conclusão, buscando compreender e analisar os fenômenos testemunhados, uma observação minuciosa sobre os resultados dos exames será realizada e manifestada. No primeiro momento, os elementos da estrutura de testes montada são esclarecidos e a justificativa pela preferência de um *framework* *open source* é exposta. Depois, os outros agentes participantes nos testes são apresentados e uma primeira bateria de experimentos é efetuada com o agente simplificado, pormenorizando seu desempenho. O mesmo procedimento é então executado com a versão completa da inteligência artificial proposta. Ainda, ambos os agentes enfrentarão adversários humanos com níveis de habilidade distintos.

Como o agente proposto é dedicado aos jogos de Texas Hold’em sem limite de apostas e com apenas seis competidores, as situações de testes deverão simular identicamente o ambiente de atuação pretendido. Quando necessário o enfrentamento direto entre somente duas entidades artificiais, múltiplas instâncias do mesmo agente serão alocadas na mesa para conciliar com o padrão de regras especificado. Por exemplo, em um experimento entre os dois agentes construídos, a configuração de mesa intercalaria a posição dos mesmos repetidamente (ReduxBot/EssentialBot/ReduxBot/EssentialBot/ReduxBot/EssentialBot). Depois, para determinar os resultados, os ganhos de cada agente são somados com os de seus congêneres e divididos entre o número total de instâncias de agentes idênticos.

Para compreender satisfatoriamente os resultados provindos dos experimentos e testes entre os agentes, é necessário primeiramente assimilar as convenções de lucro e desempenho estabelecidas pela comunidade acadêmica do pôquer. A presença de elementos estocásticos permite que um jogador obtenha margens de lucro além das expectativas ou que tenha prejuízos mesmo jogando de maneira teoricamente perfeita. Desse modo, as conclusões resultantes da análise dos resultados devem considerar as consequências do não-determinismo intrínseco ao jogo.

De acordo com Johanson (2007), a variância presente no pôquer é de aproximadamente ± seis *big blinds* por mão. Aplicando as técnicas de permutação de lugares, explicadas na seção 3.3, é possível reduzir a influência estocástica para ± 1.6 (um ponto seis) *big blinds* por mão. Ainda que esses valores não aparentem ser expressivos, sua significância é contextualiza ao se analisar os ganhos comumente auferidos pelos jogadores habituais. Por exemplo, um ótimo jogador consegue sustentar uma margem de lucro de cinco *big blinds* a cada cem mãos (THOMAS, 2008). Portanto, objetivando reduzir as consequências da variância nos testes idealizados, todas as simulações empregarão o algoritmo de permutação de lugares e executarão um número considerável de mãos. Para distinguir com exatidão os méritos de cada agente, um único baralho pré-serializado será construído e utilizado. Qualquer variação nessas configurações será explicitamente especificada.

Por fim, os resultados das simulações serão ilustrados através de gráficos, tabelas e quadros, estando sempre acompanhados de uma breve explicação sobre as circunstâncias que sucederam os mesmos. Todos os históricos de mãos gerados durante esse estágio serão posteriormente exportados para um *software* de estatística de pôquer e as informações qualitativas e quantitativas de cada agente desenvolvido serão decifradas.

## Ambiente de testes

A correta avaliação do desempenho dos agentes depende exclusivamente de um ambiente de testes confiável, preciso e à prova de falhas. As funções de um *framework* competente incluem a criação de um padrão de funcionamento cujos agentes devem resignar-se, a simulação da estrutura completa de jogo e a coleta dos dados subsequentes aos experimentos. Desse modo, é evidente a preferência por não desenvolver um ambiente de testes do princípio e, ao contrário, preterir a uma implementação bem conceituada da API Meerkat. A lista abaixo apresenta os principais motivos atinentes a essa decisão.

* Disponibilidade: a interface de programação Meerkat é disponibilizada publicamente para todos interessados na construção de agentes para pôquer. Ainda, uma comunidade ativa de pesquisadores reunem-se compartilhando conhecimento sobre o domínio e publicando seus próprios agentes para testes.
* Abrangência: devido ao seu grande reconhecimento, a API é freqüentemente empregada por diversos estudiosos e grupos especializados para a criaçao de agentes. Essa acessibilidade, além de promover o estudo da área, impulsiona o desenvolvimento do domínio.
* Redundância: do ponto de vista computacional, a concepção de um *framework* para pôquer é um procedimento complexo e dispendioso. Portanto, mesmo com os benefícios de aprendizado, essa tarefa não agregaria valor ao âmbito primordial do projeto.

Uma breve pesquisa foi efetuada para encontrar o ambiente de testes ideal. As subseções seguintes apresentam as principais alternativas encontradas e escolhidas para realizar essa tarefa. Após a introdução, são mencionados os pontos positivos e negativos de cada opção. Todos os *frameworks* listados estão em conformidade com os ítens referidos acima.

### PokerAcademy

O PokerAcademy é uma implementação proprietária da API Meerkat que, além de permitir que o agente interaja com seus congêneres, possibilita que o próprio desenvolvedor enfrente sua criação (POKERACADEMY, 2010). Por outro lado, como essa ferramenta foi desenvolvida com intuito de auxiliar jogadores principiantes no aprendizado do pôquer e não com o objetivo único de ser uma plataforma de testes para agentes, a velocidade das simulações é relativamente lenta. Parte disso deve-se a existência de uma interface gráfica elaborada que demonstra explicitamente as ações de cada agente ao usuário.

Porém, como o foco dos criadores desse framework – os pesquisadores da Universidade de Alberta – sempre direcionou-se ao pôquer com limite de apostas, não há uma gama muito atraente de agentes referentes ao jogo estudado nesse trabalho. Ainda assim, é possível configurar o programa para utilizar qualquer agente em concordância com os padrões de programação da API Meerkat.

### Open Meerkat Testbed

Consequentemente, em face das dificuldades presentes no PokerAcademy, um grupo de programadores desenvolveu uma implementação *open source* da *API Meerkat*­4, objetivando maior agilidade nas simulações. Esse projeto, apesar de recente, já dispõe dos meios necessários para a execução de centenas de mãos de pôquer por segundo, um avanço bastante significativo. Ainda, os históricos de todas as mãos são arquivados em um formato compatível com os softwares de estatísticas. Essa integração entre o *framework* e outros programas, como o *PokerTracker* e o *PokerAcademy*, é um grande diferencial vantajoso e auxilia em diversos aspectos no desenvolvimento de agentes.

Primeiramente, não há necessidade de criar um programa para conversão dos históricos de mão. Isso permite que as mãos sejam exportadas tão logo estejam concluídas e que os agentes consultem essas informações imediatamente, resultando em uma capacidade adaptativa ágil. Segundo, a relação de atomicidade entre os ambientes de testes faz com que não seja exigido nenhum esforço posterior para conformar os agentes aos padrões impostos. Como os *frameworks* apresentam qualidades complementares, a rapidez de simulação e a possibilidade de observação/interação com os agentes, a redundância de execução estimula testes mais eficazes.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4 http://code.google.com/p/opentestbed/

Outra característica relevante do *Open Meerkat Testbed* é que seu código está disponível publicamente e seus autores incentivam a participação dos usuários no desenvolvimento ativo da plataforma. Assim, além de permitir que sejam realizadas mudanças para aprimorar a performance dos testes, o produto ainda acompanha quatro agentes para Texas Hold’em pôquer sem limite de apostas e alguns algoritmos indispensáveis para a construção de programas inteligentes para o jogo, como métodos para avaliar a qualidade de uma determinada mão.

### Oponentes

A elaboração de agentes para pôquer é infundada quando não existem adversários, sejam artificiais ou verdadeiros, capazes de desafiar as proposições apresentadas por seu autor. De certa forma, é a própria existência dos mesmos que serve de motivação à área, haja vista que é o referencial competitivo que determina o patamar de desempenho almejado. O interesse original dos pesquisadores no jogo de xadrez é um reflexo transparente desse comportamento.

Com base nas ideias supracitadas, os agentes selecionados para participar dos experimentos compartilham algumas características essenciais para o exame, análise e homologação das entidades computacionais desenvolvidas nesse projeto. Essas propriedades, embora não correspondam obrigatoriamente na competência dos adversários no jogo, tem o objetivo de confirmar a capacidade adaptativa e explorativa dos agentes. A lista abaixo denomina todos participantes presentes nos testes e conjectura o seu caráter de utilidade nas simulações.

* AlwaysCallBot – como o nome indica, é um agente arcaico que paga toda e qualquer aposta, independente das cartas que possui ou da situação em que está inserido. Sua proficuidade reside exatamente na sua incapacidade hábil, considerando que a performance dos seus adversários será diretamente proporcional a capacidade de adaptação aos seus defeitos. Ou seja, qualquer agente minimamente inteligente é capaz de vencê-lo, mas a margem da vitória é que realmente confirmará sua qualidade.
* FlockBot – uma modelagem simplista dos oponentes reais encontrados nos níveis inferiores do pôquer online. Jogam quantidades demasiadas de mão, são extremamente passivos e freqüentemente não desistem até que todas as cartas comunitárias sejam reveladas.
* ChumpBot – versão aperfeiçoada do FlockBot, com um grau de imprevisibilidade maior e um estilo de jogo mais agressivo – inclusive com blefes esporádicos em seu repertório de jogadas. Ainda assim, o agente não desiste suficientemente às apostas dos adversários e o valor dos seus *raises* coincide com a qualidade da mão que possui.
* SimpleBot – é o mais complexo entre os agentes já descritos. Assemelha-se com o ReduxBot em diversos aspectos, sendo também um sistema híbrido baseado em regras e fórmulas. No entanto, a refinação pré-flop é drasticamente simplista e não considera conceitos essenciais como posição e deceptividade. Nas rodadas posteriores, contrariamente, o agente usa os algoritmos de avaliação de mão da Universidade de Alberta e mostra-se consistente.
* Anti-Redux – uma inteligência artificial projetada apenas para explorar as falhas no ReduxBot, especialmente sua inaptidão ao enfrentar agressividade no pré-flop. Essa postura hostil é mantida também no pós-flop e o agente efetua blefes, semi-blefes e outras manobras.
* ORenegado – a primeira implementação do ReduxBot, cujas premissas estratégicas não evoluíram suficientemente para serem documentadas no projeto. Desse modo, os resquícios do agente são utilizados para auxiliar o processo de testes do seu descendente.

## Agente simples

Conforme mencionado anteriormente, a rodada inaugural de testes será protagonizada pelo primeiro agente apresentado, o ReduxBot. Pretende-se, durante o curso desses experimentos, observar o comportamento do agente em circunstâncias distintas e analisar as competências de um sistema baseado em regras. Os testes em questão foram projetados para situar o agente em condições extremas, visando examinar seu funcionamento sob situações vertiginosamente vantajosas e prejudiciais. Assim, será possível determinar com clareza os méritos e deméritos do programa concebido.

### Situação genérica

A primeira circunstância produzida pretende reproduzir uma situação de jogo genérica, com oponentes construídos sem nenhum propósito especial de exploração ou métrica de desempenho. Nesse caso, deseja-se observar como um agente sem modelagem de oponentes se comporta em um ambiente de jogo comum. O gráfico a seguir ilustra o desempenho dos participantes do primeiro teste.

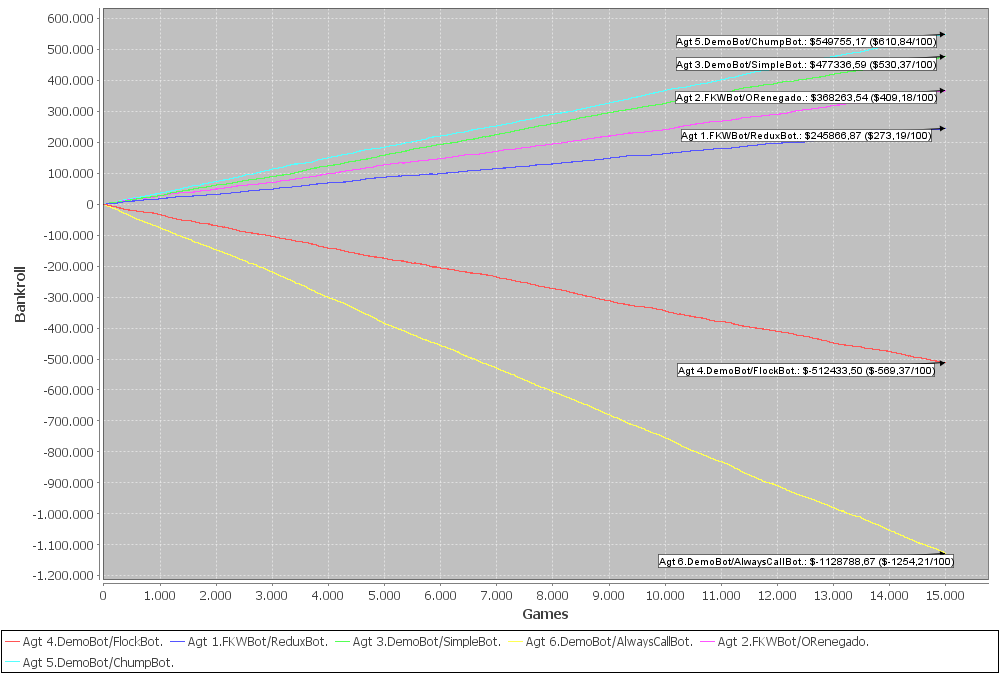


Gráfico 1 ReduxBot em situação genérica

Fonte: Elaborado pelo autor

No gráfico acima, pode-se perceber um padrão que será evidenciado na maioria dos experimentos realizados: os agentes superiores se aproveitando incessantemente dos mais fracos. Há uma diferença substancial entre as margens dos vencedores e um prejuízo notório do agente de menor qualidade. Mesmo sem modelagem de oponentes, o ReduxBot pode auferir ganhos expressivos e, mesmo que inconscientemente, explorar seus adversários menos qualificados de forma adequada. Para demonstrar essa afirmação com mais clareza, a próxima tabela revela a performance do mesmo contra cada um dos adversários presentes.

O caráter elucidativo da tabela permite visualizar alguns elementos interessantes do jogo e que representam fielmente o desafio proposto pelo pôquer. Primeiramente, ORenegado, mesmo sendo uma versão primitiva do ReduxBot, é capaz de vencê-lo com lucro bastante reduzido e é a única entidade que aufere ganhos sobre todos os competidores. Como a quantidade de mãos executadas não é suficiente para remover o efeito da variância no jogo, experimentos com apenas os dois agentes serão realizados posteriormente. Ainda, o ChumpBot é o agente com melhor classificação financeira mesmo tendo prejuízos contra três outros participantes. Isso acontece devido a sua semelhança pré-flop com os dois piores jogadores, FlockBot e AlwaysCallBot, que inclui-o em uma grande quantidade de mãos contra os dois agentes referidos. Assim, como é mais habilidoso nas rodadas subsequentes, o mesmo é capaz de auferir grandes vencimentos.

Tabela 2 Resultados do ReduxBot em situação genérica

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ReduxBot | ORenegado | SimpleBot | AlwaysCall | ChumpBot | FlockBot |
| ReduxBot |  | -$1,135 | $2,151 | $118,106 | $19,545 | $106,639 |
| ORenegado | $1,135 |  | $2,186 | $171,157 | $35,884 | $155,762 |
| SimpleBot | -$2,151 | -$2,186 |  | $271,009 | $53,824 | $152,728 |
| AlwaysCall | -$118,106 | -$171,157 | -$271,009 |  | -$436,540 | -$129,358 |
| ChumpBot | -$19,545 | -$35,884 | -$53,824 | $436,540 |  | $213,976 |
| FlockBot | -$106,639 | -$155,762 | -$152,728 | $129,358 | -$213,976 |  |

### ReduxBot *versus* ORenegado

Quando projetada uma situação genérica, o agente ORenegado esteve apto a sobrepujar sua versão mais elaborada tanto no embate individual quanto nos ganhos gerais contra os outros competidores. As principais diferenças entre ambos situam-se no modelo pós-flop. O ReduxBot, ao contrário do seu antecessor, consulta o curso de ação realizado no pré-flop para influenciar em sua decisão e, portanto, possui maior probabilidade de ser agressivo se já assumiu essa postura nas rodadas anteriores. Ainda, o ORenegado infere o mesmo processo decisório para todas as rodadas de aposta enquanto que seu adversário assume formas de jogo distintas de acordo com a rodada atual do jogo, necessitando de mãos mais poderosas para inferir apostas na medida em que o jogo avança.

Agora, a simulação propõe uma mesa com três instâncias de cada agente, visando testemunhar qual agente é de fato superior. Para isso, serão executadas cento e oitenta mil mãos em seis permutações de trinta mil. A soma dos lucros de cada agente congênere demonstrará os resultados auferidos por sua respectiva classe. O gráfico a seguir demonstra os resultados de cada entidade nesse experimento. Com um número de mãos maior que no teste genérico, a imagem ilustra claramente o domínio do ReduxBot sobre seu oponente. Em conclusão, é possível afirmar que os pesquisadores devem estar atentos antes de realizarem qualquer afirmação precipitada em ambientes com elementos estocásticos.

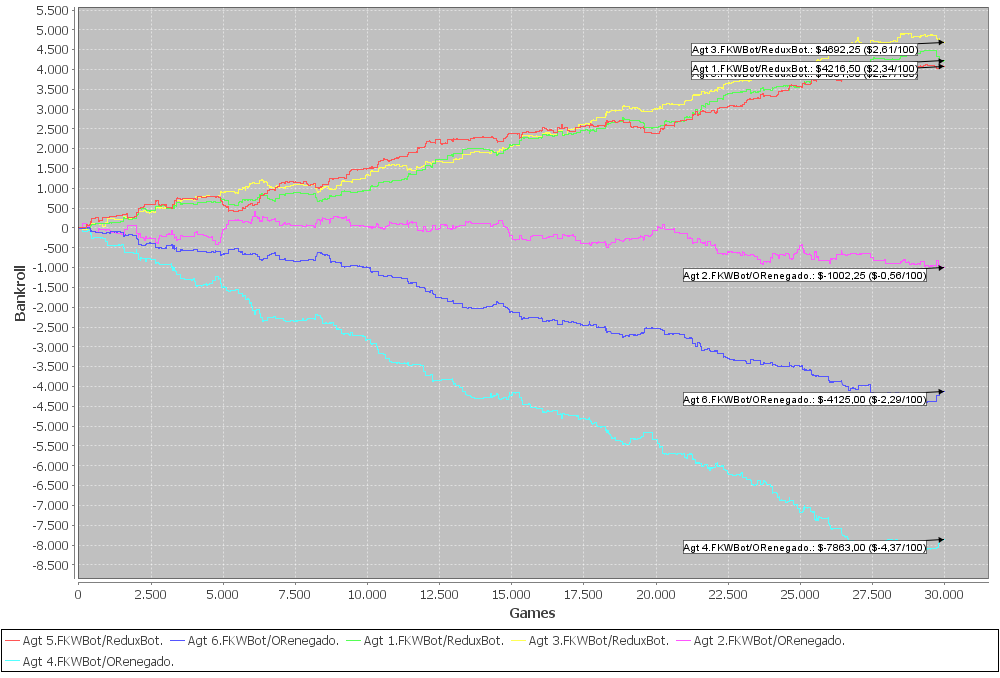


Gráfico 2 Enfrentamento Reduxbot versus ORenegado

Fonte: Elaborado pelo autor

### Oponente explorador

Ao enfrentar oponentes com arquiteturas similares a sua, exclusivamente baseadas em regras, o ReduxBot obteve um desempenho favorável. Como não integram um componente de modelagem de oponentes, todos esses agentes não identificam e, conseqüentemente, não exploram as fraquezas presentes no agente implementado, e vice-versa. Logo, mesmo um competidor capaz de vencer todos os seus oponentes não possui nenhuma garantia que será o maior ganhador da mesa, como demonstrado no primeiro experimento realizado. Similarmente, a inconsciência inerente a esses sistemas obriga-os a conceber novas regras sempre que uma nova situação é vivenciada, originando uma predisposição a prejuízos se um agente específico apresenta condições de distinguir os traços característicos dessas entidades e adaptar-se convenientemente.

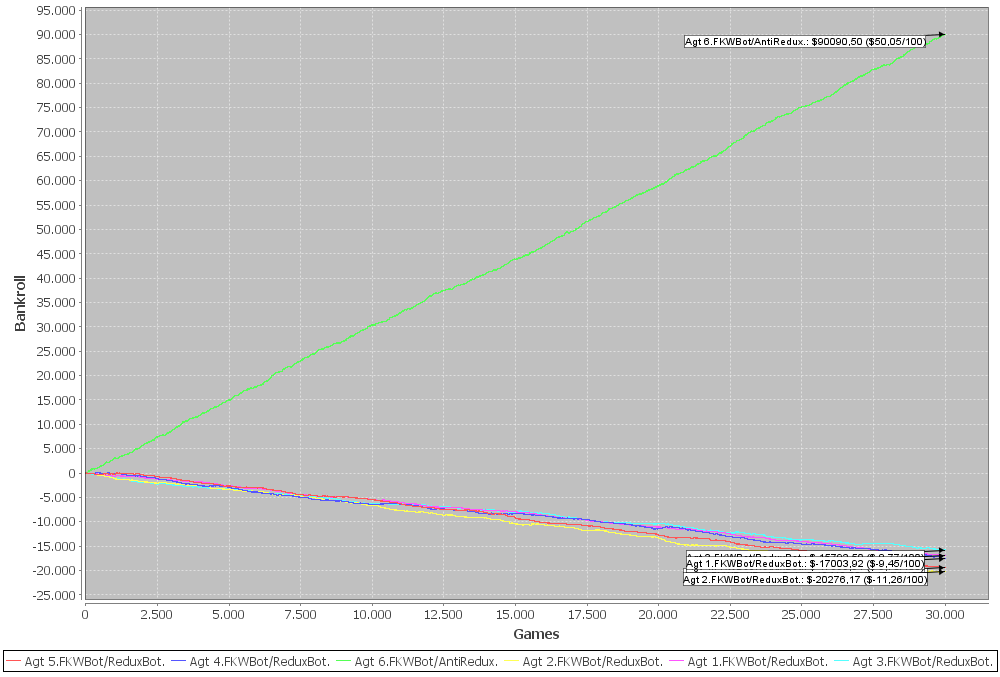


Gráfico 3 ReduxBot contra um oponente explorador

Fonte: Elaborado pelo autor

Embora nenhum dos agentes disponíveis esteja apto a evidenciar os argumentos supracitados (com exceção do EssentialBot, que será apresentado posteriormente), um sistema baseado em regras cujo único propósito reside em explorar a inépcia do ReduxBot enfrentará cinco instância do mesmo durante cento e oitenta mil mãos. A imagem acima demonstra os resultados dessas partidas e comprova as afirmações anteriores. Conclusivamente, a única maneira de interromper esse prognóstico seria através da intervenção humana e a construção de novos conjuntos de regras que removessem essa falha protuberante no agente testado

### Oponente previsível

Em uma última circunstância de teste, o ReduxBot é introduzido em um ambiente amistoso contra um agente desqualificado e amplamente previsível. O AlwaysCallBot, conforme detalhado anteriormente, resume-se a uma única e repetida ação, pagando todas as opostas que são inferidas. Excepcionalmente, esse último experimento alocará apenas uma posição da mesa ao jogador adversário, mantendo cinco entidades do agente testado. As razões dessa preferência serão demonstradas quando o mesmo teste for realizado com o agente completo. Foram executadas sessenta mil mãos, com uma nova permutação a cada dez mil jogos.

Apenas a filtragem de mãos executada no pré-flop por parte do agente construído deve garantir uma vantagem confortável contra seu adversário (que joga todas as mãos que lhe são distribuídas). Porém, nas rodadas conseguintes, o ReduxBot executa seu modelo único e determinístico para selecionar o curso de ação a ser tomado. Como essa abordagem é baseada na comparação da avaliação da mão com um limiar pré-estabelecido, o agente é incapaz de distinguir que seu oponente está propenso a possuir mãos muito mais fracas que o normal e adaptar esse limiar de acordo. Assim, o mesmo estilo de jogo é praticado contra todos os competidores, independente de suas particularidades.

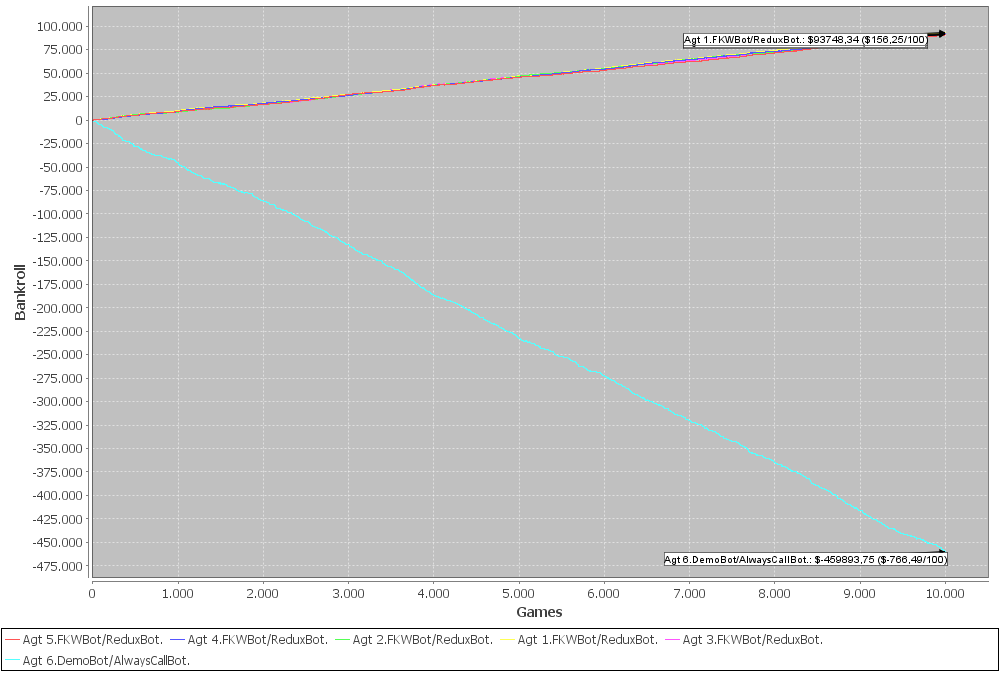


Gráfico 4 ReduxBot versus AlwaysCallBot

Fonte: Elaborado pelo autor

A imagem anterior ilustra o desempenho do agente na situação projetada. O aspecto importante desse experimento consiste em examinar o grau de exploração perpetrado pelo sistema artificial. Como opta por não participar da maioria das partidas, exceto quando possui cartas privadas qualificadas, a probabilidade de vitória do agente será sempre superior a de um adversário que habita todos os jogos. Porém, inúmeras vezes, o sistema abstém-se de apostar imaginando que suas cartas não são fortes insuficientes, mesmo que sejam bastante superiores ao conjunto possuído pelo adversário (que consiste de todas as mãos possíveis).

## Agente completo

Como principal produto desse projeto, o agente completo passará por um número maior de experimentações que seu antecessor, bem como seus testes serão mais extenuantes. Assim sendo, nem todos serão apresentados graficamente. Algumas situações projetadas são idênticas as impostas ao ReduxBot, objetivando distinguir o comportamento de ambos, e outras foram idealizadas com intuito de verificar a capacidade de adaptação do agente e sua perícia na exploração dos adversários. As seções seguintes detalham cada um dos testes executados na homologação do sistema.

### Situação Genérica

A primeira circunstância incorporará o mesmo ambiente concebido no primeiro teste do ReduxBot. Com os mesmos adversários, o mesmo número de mãos e exatamente as mesmas cartas sendo distribuídas durante as rodadas, é possível clarificar a diferença de habilidade rapidamente e estabelecer conclusivamente a superioridade entre os dois agentes construídos. Depois, será possível deliberar sobre os motivos que consolidam a disparidade de funcionamento. O Gráfico 5 ilustra os resultados da situação genérica.

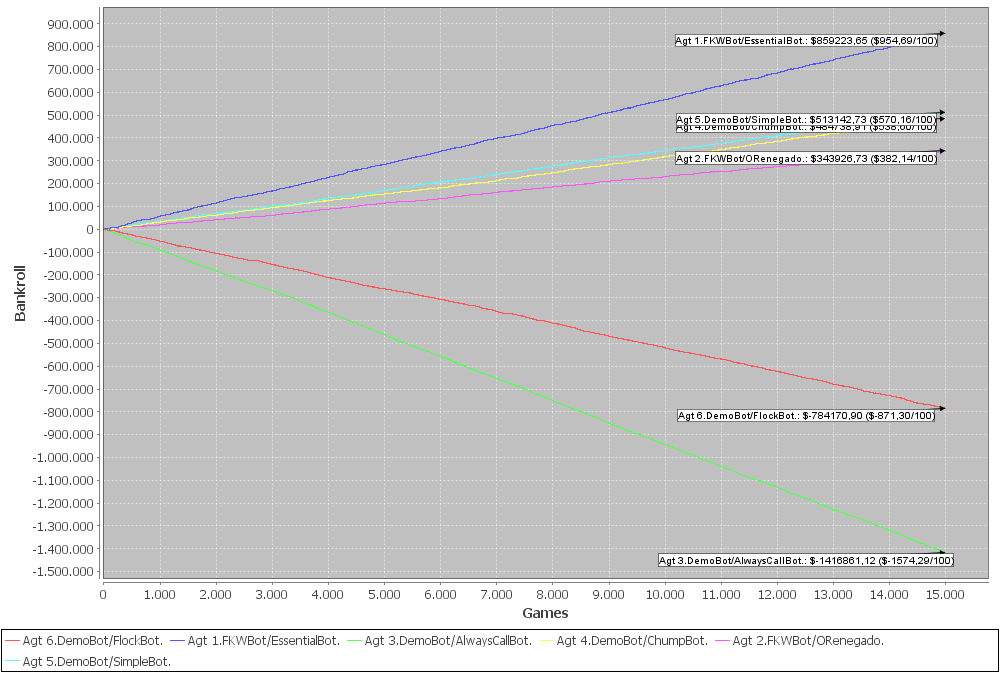


Gráfico 5 EssentialBot em situação genérica

Fonte: Elaborado pelo autor

Primeiramente, nota-se que os dois piores agentes apresentam desempenhos bastante inferiores aos obtidos nos testes genéricos contra o agente simples, com margens de prejuízo cerca de trinta por cento superiores. Em oposição, do lado oposto do gráfico, os adversários vencedores, que antes apresentavam intervalos proporcionais entre os lucros auferidos, se encontram aglomerados em um ponto próximo. Nessa perspectiva, agente proposto pelo projeto destaca-se pelos ganhos extremamente elevados, quase dobrando os valores logrados por seu melhor competidor. Entretanto, apesar de intuitivos, os gráficos geralmente escondem algumas características particulares do jogo, mostrando os resultados de uma visão macroscópica. A Tabela 3 traça os valores angariados por cada um dos agentes contra a totalidade de seus competidores.

Tabela 3 Resultados do EssentialBot em situação genérica

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | EssentialBot | ORenegado | SimpleBot | ChumpBot | FlockBot | AlwaysCall |
| EssentialBot |  | -$1,481 | -$10,492 | $45,777 | $361,955 | $449,846 |
| ORenegado | $1,481 |  | $4,903 | $30,077 | $146,447 | $161,170 |
| SimpleBot | $10,942 | -$4,903 |  | $52,701 | $177,786 | $271,116 |
| ChumpBot | -$45,777 | -$30,077 | -$52,701 |  | $206,722 | $399,801 |
| FlockBot | -$361,955 | -$146,447 | -$177,786 | -$206,722 |  | $129,953 |
| AlwaysCall | -$449,846 | -$161,170 | -$271,116 | -$399,801 | -$129,953 |  |

Um aspecto interessante desse experimento é que, mesmo com uma vantagem confortável no âmbito geral, o agente não obteve um desempenho superior contra todos os adversários. Os dois agentes mais qualificados, ORenegado e SimpleBot, mesmo não vencendo significativamente, estiveram aptos a conquistar vitórias pessoais sobre o agente projetado. A explicação para esse acontecimento reside na polaridade do estilo de jogo dos competidores presentes. Enquanto que os adversários fracos não realizam qualquer filtragem de mãos no pré-flop, os jogadores mais competentes estabelecem um conjunto seleto de mãos propícias a continuar no jogo. Essa inconsistência obriga que o EssentialBot gere uma estratégia pré-flop dinâmica que concilie as particularidades de ambas as categorias, em um processo de hibridação, aderindo, assim, um estilo de jogo equilibrado para si.

Como as características dos adversários são opostas, é natural que o agente idealize uma estratégia harmoniosa e centralizada, objetivando a exploração de todas as classes de competidores. Porém, na medida em que adere uma gama maior de mãos em sua estratégia, o agente torna-se vulnerável contra oponentes que realizem uma filtragem pré-flop intensiva. Apenas para fins comparativos, após o experimento documentado, o EssentialBot apresentou um VPIP de 49.6 (quarenta e nove ponto seis), enquanto que o ORenegado registrou um índice de 16.1 da mesma estatística. Ou seja, o agente completo participa de três vezes mais mãos que esse rival específico e, desse modo, possui – regularmente – cartas privadas proporcionalmente inferiores, tendo que reverter essa desvantagem competitiva nas rodadas conseguintes.

### EssentialBot *versus* ReduxBot

Durante a conceitualização e desenvolvimento dos agentes propostos, os méritos de cada abordagem foram esclarecidos, bem como a motivação para seu estudo. Mesmo assim, embora a metodologia de construção de cada agente distancie-se em alguns pontos decisivos, gerando características conspícuas em cada entidade, outros conceitos mantiveram-se presentes em ambas arquiteturas, como a própria constituição da estrutura dos sistemas. Portanto, embora partam de premissas divergentes, há, inevitavelmente, concentricidade em sua composição.

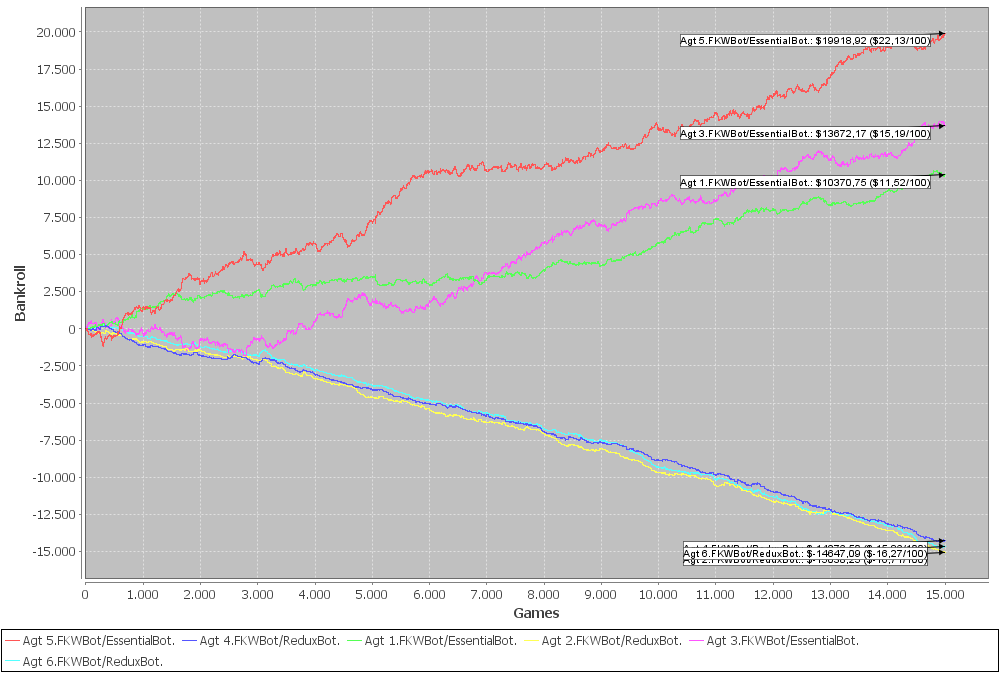


Gráfico 6 EssentialBot versus ReduxBot

Fonte: Elaborado pelo autor

Entretanto, a existência de fatores compartilhados não assegura a similaridade de execução, sendo os próprios testes uma arguição dessa asserção. Agora, mais do que determinar o agente superior, as experimentações ilustradas no gráfico anterior propõem identificar e autenticar as evidências do comportamento explorativo projetado. Nesse sentido, além da disputa individual entre os dois agentes construídos no curso desse projeto, a próxima circunstância de testes será também aplicada para analisar os históricos de mãos desses confrontos e encontrar testemunhos verídicos de exploração e adaptação ao oponente. Devido à impossibilidade de transcrever essas ideias, anexou-se uma pequena quantidade de históricos de mãos comprovatórias ao fim do trabalho, onde é possível perceber claramente uma extração de lucros em cima das duas principais debilidades do ReduxBot: a extrema filtragem pré-flop e a incapacidade de blefe (ver Anexo I).

### Embates individuais

Além do enfrentamento anterior, julgou-se necessário demonstrar o desempenho do agente em embates individuais contra todos os outros adversários disponíveis. Assim como nos outros confrontos, a configuração de mesa contém três instâncias de cada entidade e o número de mãos executadas é de noventa mil, com seis permutações de quinze mil. No entanto, ao invés de mensurar os ganhos através do total auferido, os mesmos serão demonstrados em *big blinds* a cada cem mãos. Essa métrica é empregada pois não é proporcional ao número de mãos efetuada. Por exemplo, se um agente consistentemente ganha um *big blind* por mão, seus resultados em *BB/100 (big blinds per hundred)* serão o mesmo independente da amostragem, enquanto que seus lucros ampliam linearmente. Nos gráficos elucidados anteriormente, esses índices podem ser consultados entre parênteses ao lado do nome dos agentes. A tabela abaixo mostra os resultados do EssentialBot em cada experimento realizados.

Tabela 4 Ganhos do EssentialBot em cada confronto individual

|  |  |  |
| --- | --- | --- |
| Oponente | | |
| ORenegado | ChumpBot | SimpleBot |
| 18,41 BB/100 | 135,26 BB/100 | 1,10 BB/100 |

### Agente previsível

O último teste proposto consiste em analisar o desempenho e o comportamento do agente construído contra um oponente absolutamente previsível, conforme realizado nos experimentos do ReduxBot. Semelhantemente, o mesmo agente desqualificado previamente empregado protagonizará essa simulação para manter a consistência dos resultados. Uma observação especial sobre essa circunstância de teste baseia-se no fato que o EssentialBot gera uma árvore de jogo para cada decisão que precisa realizar. Com isso, contra um oponente específico que nunca desiste, é preferível inserir apenas um adversário e cinco instâncias do próprio agente. Do contrário, é computacionalmente inviável realizar um número satisfatório de mãos experimentais, visto que todas as entidades do agente adversário permaneceriam até o fim da mão e as árvores construídas seriam muito mais extensas. O gráfico abaixo ilustra os resultados obtidos.

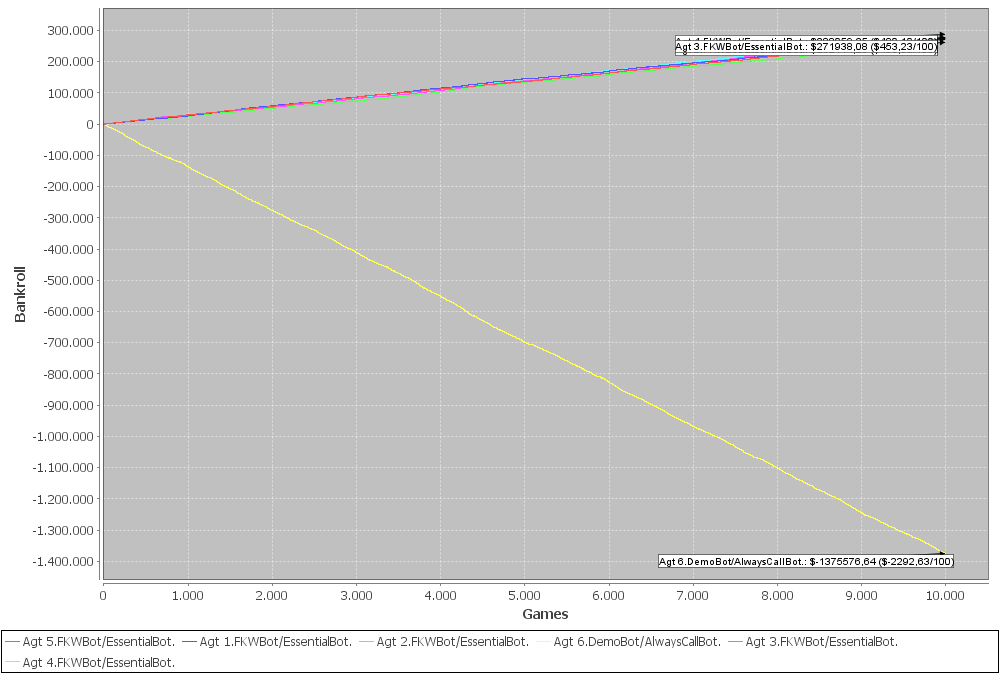


Gráfico 7 EssentialBot *versus* AlwaysCallBot

Fonte: Elaborado pelo autor

Em uma breve comparação quantitativa entre os ganhos auferidos pelos dois agentes idealizados nesse trabalho, já é possível perceber um nível de jogo indubitavelmente superior por parte do agente complexo. Nessa seção de testes, o AlwaysCallBot obteve um prejuízo três vezes superior do que quando enfrentara o agente simples. Porém, a correta avaliação das habilidades cognitivas do agente necessita exame de variáveis qualitativas, como o grau de adaptação inferido pelo próprio. Nesse sentido, a admissão de um agente extremamente limitado concede meios propícios de identificar um comportamento ideal. Por exemplo, contra um adversário que paga todas as apostas, independente das cartas privadas que possui, é intuitivamente equivalente assumir uma contra-estratégia que aposte com um conjunto de mãos muito mais fracas, visto que é mais provável que oponente possua uma mão ruim. Para visualizar esse conceito graficamente, o próximo gráfico manifesta a distribuição de um histograma de força de mãos do AlwaysCallBot em uma situação comum.

Gráfico 8 Histograma de frequência

Fonte: Elaborado pelo autor

Firmando-se nos dados apresentados, e ressalta-se que o agente é constituído de diversos histogramas semelhantes a esse, é concebível estimar a utilidade de uma jogada com facilidade, já que todas as variáveis requisitadas para o cálculo estão disponíveis ou podem ser rapidamente consultadas. Ainda, como o AlwaysCallBot joga atomicamente, é sensato que seu histograma esteja proporcionalmente distribuído. Por fim, como efetuado em experimentos anteriores, algumas mãos que apresentam um comportamento inteligente por parte do agente estão anexadas (ver Anexo II). Essas amostras foram selecionadas com base nas características supracitadas, onde o agente infere apostas com mãos consideradas frágeis e em situações onde a melhor decisão não é evidente e distinguível.

CONCLUSÃO

Há muito tempo, quando a inteligência artificial começava a ser estudada e desmistificada, apenas os jogos determinísticos com informações perfeitas eram estudados. O panorama favorável das contribuições iniciais criava expectativas exageradas sobre as tecnologias futuras, inclusive entre os estudiosos da área. No entanto, passado o entusiasmo inaugural, a corrente de falsas pretensões sobre a fertilidade do campo fora estagnada, permitindo a correta avaliação das limitações dessas pesquisas. Então, embora os jogos com informações perfeitas sejam costumeiramente focados para o estudo da área, criava-se a necessidade de expandir as pesquisas para domínios mais realistas.

Assim, em progressão natural, os jogos com informações imperfeitas e elementos estocásticos passam a ser evidenciados cada vez mais. Além de possuírem propriedades com considerável motivação científica, como a aleatoriedade e a insuficiência de dados para a tomada de decisão, as contribuições e descobertas advindas dessa classe de jogos possui maior aplicabilidade em situações reais. O pôquer, em especial, agrega outros obstáculos fascinantes e desafiadores na resolução do problema: múltiplos agentes, variância nos resultados, indução a erros e modelagem de oponentes.

Portanto, em concordância com as tendências e os méritos científicos expostos, a fase inicial deste trabalho discorreu sobre os fundamentos imprescindíveis para o desenvolvimento de um agente racional para Texas Hold’em, na sua variante sem limite de apostas. As questões estruturais apresentadas no primeiro capítulo, que referenciam as abordagens arquitetônicas habitualmente empregadas nas aplicações de agentes inteligentes, em conjunto com os conceitos essenciais sobre o jogo de pôquer revelados na segunda seção do trabalho, fornecem a motivação para o estudo histórico dos agentes no pôquer, presente na terceira e última parte do trabalho. Desse modo, torna-se conveniente investigar as experiências passadas de cientistas com finalidades semelhantes, buscando identificar os elementos críticos de suas jornadas.

Ainda, o conhecimento científico pesquisado e referido neste trabalho constitui grande parte do embasamento teórico necessário para a compreensão dos tópicos cruciais no desenvolvimento de um programa inteligente para pôquer, como introduzido no quarto capítulo. Nessa fase conclusiva, os conceitos apresentados são empregados na construção de um agente racional complexo que engloba uma estrutura de dados avançadas para modelagem de oponentes.

Entre as propriedades mais interessantes presentes no domínio estudado estão a sua enorme extensão e sua complexidade inerente. Com isso, o processo de mapeamento de uma árvore completa necessita de abstrações que reduzam a dimensão do ambiente e, paralelamente, mantenham as características conspícuas do mesmo. Assim, visando atingir esse objetivo, inúmeras técnicas de abstração foram originadas recentemente em pesquisas da área, como a abstração por ação, por quantidade de apostas e por valor final do *pot*. Para o EssentialBot, conceitualizou-se um conjunto de procedimentos denominado abstração por pontos de ação, que considera as decisões efetuadas pelos agentes, bem como associa índices de força para cada gênero de ação. Com isso, ao invés de usufruir apenas informações quantitativas para distinguir os contextos, cria-se um método qualitativo capaz de particularizar as circunstâncias de acordo com sua natureza. Mais do que isso, o propósito fundamental de qualquer técnica de abstração deve ser de reduzir vertiginosamente o domínio sem descaracterizá-lo, aproveitando-se da situacionalidade do ambiente para segregar circunstâncias, como realizado no trabalho através de índices de textura da mesa.

A modelagem de oponentes, que é amplamente aprofundada no curso do projeto, predispõe meios para identificar tendências e compreender as individualidades dos adversários. Combinando as técnicas de abstração com uma estrutura de árvore, onde informações frequências sobre as cartas privadas dos oponentes são acrescentadas em cada nodo, apresenta-se uma maneira confiável de se armazenar as particularidades do jogo em conjunção das particularidades de cada oponente, transformando o desafio da modelagem de oponentes em um problema habitual de busca – frequentemente estudados pelo campo da inteligência artificial. Outro problema corriqueiro dos sistemas de modelagem abrange a atuação do mecanismo sem uma amostragem considerável. Ainda, a alta rotatividade do pôquer obriga que a adaptação seja tão rápida quanto possível. Para restringir o impacto desse distúrbio, o agente proposto perfila os adversários já conhecidos empregando seus índices probabilísticos. Logo, quando um novo oponente é introduzido, o sistema pode procurar por um adversário antigo semelhante e consultar os dados do mesmo até que seja atingida a convergência com o novo competidor.

Outro mérito dessa pesquisa, especialmente por tratar-se da modalidade sem limite de apostas do jogo, é que todos os conceitos apresentados poderiam ser transportados para outro domínio e até utilizados em situações reais. Ou seja, desde que os valores absolutos e situacionais desse domínio sejam claros, é claramente viável implantar um agente semelhante no mesmo. Por exemplo, em situações de competitividade comercial, onde as empresas disputam desenfreadamente para proporcionar preços vantajosos aos seus clientes, um agente poderia compreender os aspectos que envolvem a alteração de preço por parte de um concorrente, as situações que esses fenômenos ocorrem e os estados atuais e futuros do próprio ambiente. A cotação do dólar, por exemplo, envolve valores absolutos (objetivos políticos e comerciais, taxas de juros) e valores situacionais (situação comercial dos países e do mercado, riscos iminentes e o poder econômico dos cidadãos) podendo ser prevista e adaptada por um agente especialista. Obviamente, qualquer afirmação nesse sentido é prematura e meramente hipotética e um longo caminho ainda deve ser percorrido para que sistemas possam ser finalmente implantados com esse propósito.

Com relação ao desempenho do sistema, no último capítulo são projetadas circunstâncias de jogo realísticas onde é possível identificar claramente a obtenção de um comportamento explorativo e adaptativo por parte do agente. Em adição, mesmo com as abstrações introduzidas, o módulo de modelagem consegue perceber cada situação distinta unicamente e alterar seu estilo de jogo adequadamente. Outras contribuições do trabalho consistem da adaptação das fórmulas de avaliação de mão, de novas técnicas para determinar a *range* dos adversários e um algoritmo para estimar o índice de textura da mesa. Por outro lado, muitas oportunidades para trabalhos futuros ainda persistem, sendo consideradas como potenciais perspectivas de continuação dessa pesquisa. Por exemplo, o modelo ainda necessita de uma amostragem relativamente alta para convergência. Segundo, a desconexão entre os componentes pré e pós *flop* impede uma margem de lucro ainda maior e novas técnicas de abstração podem ser desenvolvidas para a criação de um modelo atômico de jogo, que concilie as informações de todas as rodadas em uma única árvore de jogo.

Por fim, conforme os objetivos iniciais do trabalho, pode-se desenvolver um agente capaz de jogar em nível avançado utilizando e extendendo muitas das técnicas propostas na bibliografia sobre o tema, como as árvores de contexto de Billings (2006) e a metodologia de avaliação de mãos de Peña (1999). Em consoante, o conceito de *range*s possui atuação implícita nos nodo de estado da árvore, em forma de histogramas, e desse modo, atinge uma proposição secundária da pesquisa – implementar técnicas frequentemente empregada por jogadores profissionais no mundo real.

REFERÊNCIAS BIBLIOGRÁFICAS

ABOURISK, N., **Using Counterfactual Regret Minimization to Create a Competitive Multiplayer Poker Agent**. 2009. 73 f. Tese (Mestrado em Ciência da Computação), University of Alberta, Edmond, Alberta, 2009.

AGRE, P; CHAPMAN, D., **Pengi**: An Implementation of a Theory of Activity. 1987.

BEATTIE, B. et al, **Pattern Classification in No-Limit Poker**: a Head-Start Evolutionary Approach. 2007.

BIGUS, J; BIGUS, J., **Constructing intelligent agents using Java.** New York: Wiley, 2001. 408p.

BILLINGS, D., **Computer Poker**. 1995. 46 f. Tese (Mestrado em Ciência da Computação), University of Alberta, Edmond, Alberta, 1995.

BILLINGS, D., **Algorithms and Assessment in Computer Poker**. 2006. 229 f. Tese (Doutorado em Filosofia), University of Alberta, Edmond, Alberta, 2006.

BILLINGS, D. et al, Poker as a Testbed for Machine Intelligence Research. **Lecture Notes in Computer Science**, London, UK, v. 1418, p. 228-238, 1998a.

BILLINGS, D. et al, Opponent Modeling in Poker. **Proceedings of the 15th National Conference on Artificial Intelligence**, Menlo Park, CA, 1998b.

BILLINGS, D. et al, **The Challenge of Poker**. 2001.

BILLINGS, D. et al, **Aproximating Game-Theoretic Optimal Strategies for Full-scale Poker**. 2003.

BILLINGS, D.; KAN, M., **A Tool for the Direct Assessment of Poker Decisions**. 2006. Disponível em <http://poker.cs.ualberta.ca/publications/divat-icgaj.pdf>. Acesso em: 8 jun. 2010.

BRADSHAW, J. M. (Ed.), **Software agents.** Menlo Park: The MIT Press, 1997. 480p.

BROOKS, R., **Intelligence Without Reason**. 1991.

CIVITA, V. (Ed.), **Jogos de cartas.** São Paulo: Editora Abril, 1978. 216p.

COHEN, P. et al, Trial by Fire: Understanding the Design Requirements for Agents in Complex Environments. **AI Magazine**, Menlo Park, CA, v. 10, n. 3, p. 34-48, 1989.

COPELAND, B. J., **The essential Turing.** Oxford: Claredon Press, 2004. 613p.

CROSON, R.; FISHMAN, P.; POPE, D., Poker Superstars: Skill or Luck. **Chance Magazine**, [S.l], v.21, n. 4, p. 25-28, dez. 2008.

DAVIDSON, A., **Using Artificial Neural Networks to Model Opponents in Texas Hold’em**. 1999. Disponível em <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=1 0.1.1.21.7393>. Acesso em: 9 nov. 2010.

DAVIDSON, A., **Opponent Modeling in Poker: Learning and Acting in a Hostile and Uncertain Environment**. 2002. 100 f. Tese (Mestrado em Ciência da Computação), University of Alberta, Edmond, Alberta, 2002.

D´INVERNO, M.; LUCK, M., **Understanding agent systems.** New York: Springer, 2001. 191p.

ETZIONI, O.; WELD, D., Intelligent Agents on the Internet: Fact, Fiction, and Forecast. **IEEE Expert: Intelligent Systems and Their Applications,** Piscataway, NJ, v. 10, n. 4, p. 44-49, ago. 1995.

FOLLEK, R., **SoarBot:** **A Rule-Based System for Playing Poker**. 2003. 75 f. Tese (Mestrado em Ciência da Computação), Pace University, New York, NY, 2003.

GENESERETH, M. R.; NILSSON, N. J., **Logical foundations of artificial intelligence.** Los Altos: Morgan Kaufman, 1987. 405p

GILPIN, A.; SANDHOLM, T., **A Competitive Texas Hold’em Poker Player via Automated Abstraction and Real-Time Equilibrium Computation**. 2006. Disponível em <http://www.cs.cmu.edu/~sandholm/texas.aaai06.pdf>. Acesso em: 22 jun. 2010.

GILPIN, A.; SANDHOLM, T.; SORENSEN, T., A Heads-Up No-Limit Texas Hold'em Poker Player: Discretized Betting Models and Automatically Generated Equilibrium-Finding Programs. **Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems** (AAMAS 2008), Estoril, Portugal, v. 2, p. 911-918, 2008.

GOODWIN, R., **Formalizing Properties of Agents**. 1993. Disponível em <http://www.agent.ai/doc/upload/200302/good93.pdf>. Acesso em: 11 jun. 2010.

GUTTMAN, R.; MOUKAS, A.; MAES, P., Agent-mediated Electronic Commerce: A Survey. **The Knowledge Engineering Review,** New York, NY, v. 13, n. 2, p. 147-159, 1998.

HAASCH, A. et al, BIRON – The Bielefeld Robot Companion. **Proc. Int. Workshop on Advances in Service Robotics**. Stuttgart, Germany, p. 27-32. 2004.

HOPGOOD, A., **Intelligent System for Engineers and Scientists**. 2. ed. Boca Raton: CRC Press, 2001. 467p.

HSU, F. H., **Behind Deep Blue: Building the Computer that Defeated the World Chess Champion**. Princeton: Princeton University Press, 2002. 320p.

HUHNS, M. N. (Ed.); SINGH M. P. (Ed.) **Readings in agents.** San Francisco: Morgan Kaufman, 1998. 523p

JENNINGS, N.; SYCARA, K.; WOOLDRIDGE, M., A Roadmap of Agent Research and Development. **Autonomous Agents and Multi-Agent Systems,** Hingham, MA, v. 1, n. 1, p. 7-38, 1998.

JENNINGS, N.; WOOLDRIDGE, M., Applications of Intelligent Agents. **Agent Technology: Foundations, Applications, and Markets,** Secaucus, NJ, p. 3-28, 1998.

JOHANSON, M., **Robust Strategies and Counter-Strategies: Building a Champion Level Computer Poker Player**. 2007. 109 f. Tese (Mestrado em Ciência da Computação), University of Alberta, Edmond, Alberta, 2007.

KAN, M., **Postgame Analysis of Poker Decisions**. 2007. 109 f. Tese (Mestrado em Ciência da Computação), University of Alberta, Edmond, Alberta, 2007.

KOLLER, P.; PFEFFER, A., Representations and Solutions for Game-Theoretic Problems. **Artificial Intelligence**, Essex, UK, v. 94, n. 1, p. 167-215, jun. 1997.

KOSH, F., **3APL-M Manual**. 2005. Disponível em <http://www.cs.uu.nl/3apl-m/docs/3aplm-manual.pdf>. Acesso em: 23 jun. 2010.

MAES, P., Artificial Life Meets Entertainment: Lifelike Autonomous Agents. **Communications of the ACM**, New York, NY, v. 38, n. 11, p. 108-114, nov. 1995.

MCCURLEY, P., **An Artificial Intelligence Agent for Texas Hold'em Poker**. 2009. Dissertação (Mestrado em Ciência da Computação), University of Newcastle Upon Tyne, Newcastle Upon Tyne, 2009.

MILLER, E.; SKLANSKY, D.; MALMUTH, M., **Small Stakes Hold'em**. Henderson: Two Plus Two Publishing, 2004. 369p.

MILLER, E.; MEHTA, S.; FLYNN, M., **Small Stakes No-Limit Hold’em**.Henderson: Two Plus Two Publishing, 2009. 376p.

MOHR, M. S., **The new games treasury.** Boston: Houghton Mifflin Company, 1997. 431p.

NASH, J., Equilibrium Points in n-Person Games. **Proceedings of the Nation Academy of Sciences**, [S.l.], v. 36, n.1, p. 48-49, jan. 1950.

OSBORNE, M.; RUBINSTEIN, A., **A Course in Game Theory**. Cambridge: The MIT Press. 1994. 368p.

PAPP, D., **Dealing with Imperfect Information in Poker**. 1998. 93 f. Tese (Mestrado em Ciência da Computação), University of Alberta, Edmond, Alberta, 1998.

PEÑA, M., **Probabilities and Simulations in Poker**. 1999. 73 f. Tese (Mestrado em Ciência da Computação), University of Alberta, Edmond, Alberta, 1999.

POKERACADEMY, **PokerAcademy Specification.** 2010. Disponível em <http://www.poker-academy.com>. Acesso em: 19 nov. 2010.

POKERTRACKER, **Poker Tracker 3 Statistical Guide.** 2010. Disponível em <http://www.pokertracker.com/products/PT3/docs/PokerTracker3\_Statistical\_Reference\_Guide.pdf>. Acesso em: 23 jun. 2010.

REZENDE, S. O. et al, **Sistemas inteligentes**. Barueri: Manole, 2005. 550p.

RUSSEL, S. J.; NORVIG, P., **Artificial intelligence: a modern approach**. Upper Saddle River: Prentice-Hall, 1995. 932p.

SAKAI, H., **Internet Poker: Data Collection and Analysis**. 2005. Disponível em <http://www.cs.brown.edu/research/pubs/theses/ugrad/2005/hsakai.pdf>. Acesso em: 21 jun. 2010.

SKLANSKY, D., **The Theory of Poker**. 3. ed. Henderson: Two Plus Two Publishing, 1994. 276p.

SOUTHEY, F. et al, Bayes’ Bluff: Opponent Modelling in Poker.2005. **Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence.** Edinburgh, Scotland, p. 550-558, jul. 2005.

STEELS, L., **Cooperation Between Distributed Agents Through Self-Organization**. 1990. Disponível em <http://dia.fi.upm.es/~phernan/AgentesInteligentes/referencias/steels90.pdf>. Acesso em: 21 jun. 2010.

THOMAS, D. Medium-Stakes Strategy: Using PokerTracker. **CardPlayer Magazine**, [S.l], v. 5, n. 4, abr. 2008.

THORP, E., **The Mathematics of Gambling.** Gambling Times, 1985. Disponível em <http://www.bjmath.com/bjmath/thorp/tog.htm>. Acesso em: 20 jun. 2010.

TUMER, K.; AGOGINO, A., **Distributed Agent-Based Air Traffic Flow Management.** 2007. Disponível em <http://web.engr.oregonstate.edu/~ktumer/ktumer-aamas07.pdf>. Acesso em: 21 jun. 2010.

VON NEUMANN, J.; MORGENSTERN, O. **Theory of Games and Economic Behavior**. London: Princeton University Press, 2007. 776p

WEISS, G. (Ed.), **Multiagent Systems**: A Modern Approach to Distributed Artificial Intelligence. London, England: The MIT Press, 1999. 643p.

WOOLDRIDGE, M.; JENNINGS, N., Intelligent Agents: Theory and Practice. **Knowledge Engineering Review**, [S.l.], v. 10, n. 2, p. 115-152,1995.

WOOLDRIDGE, M., **An introduction to multiagent systems**. New York: John Wiley, 2002. 348p.

# ANEXOS

# ANEXO I – Históricos contra reduxbot

Nessa primeira mão, o EssentialBot aproveita da insegurança de seu oponente para realizar um blefe no River com uma mão extremamente frágil. Nota-se que o ReduxBot desiste com um *overpair,* provando a eficácia das observações previamente realizadas pelo agente construído. O ReduxBot, como já mencionado, é um jogador muito precavido e opta por alcançar o *showdown* apenas com mãos poderosas, jogando sempre de forma agressiva. Logo, quando assume uma postura passiva, os adversários podem identificar que o mesmo não possui uma mão ótima. Apesar de ser um exemplo singular, que exemplifica um ponto em especial, diversas mãos semelhantes são constante encontradas nos históricos produzidos pelo embate entre ambos os agentes.

|  |
| --- |
| Game #81901603172443: Table OpenTestBed - $0.50/$1.00  Seat 1: Agt 1.FKWBot/EssentialBot. ($100.00)  Seat 2: Agt 2.FKWBot/ReduxBot. ($100.00)  Seat 3: Agt 3.FKWBot/EssentialBot. ($100.00)  Seat 4: Agt 4.FKWBot/ReduxBot. ($100.00)  Seat 5: Agt 5.FKWBot/EssentialBot. ($100.00)  Seat 6: Agt 6.FKWBot/ReduxBot. ($100.00)  Agt 1.FKWBot/EssentialBot. posts the small blind of $0.50  Agt 2.FKWBot/ReduxBot. posts the big blind of $1.00  The button is in seat #6  \*\*\* HOLE CARDS \*\*\*  Dealt to Agt 1.FKWBot/EssentialBot. [Ad 5c]  Dealt to Agt 2.FKWBot/ReduxBot. [Ks Qh]  Dealt to Agt 3.FKWBot/EssentialBot. [2h As]  Dealt to Agt 4.FKWBot/ReduxBot. [Ah 2s]  Dealt to Agt 5.FKWBot/EssentialBot. [9s 3d]  Dealt to Agt 6.FKWBot/ReduxBot. [8d 8h]  Agt 3.FKWBot/EssentialBot. raises to $3.50  Agt 4.FKWBot/ReduxBot. folds  Agt 5.FKWBot/EssentialBot. folds  Agt 6.FKWBot/ReduxBot. calls $3.50  Agt 1.FKWBot/EssentialBot. folds  Agt 2.FKWBot/ReduxBot. folds  \*\*\* FLOP \*\*\* [7d 6d 3s]  Agt 3.FKWBot/EssentialBot. bets $9.50  Agt 6.FKWBot/ReduxBot. calls $9.50  \*\*\* TURN \*\*\* [7d 6d 3s] [7h]  Agt 3.FKWBot/EssentialBot. checks  Agt 6.FKWBot/ReduxBot. checks  \*\*\* RIVER \*\*\* [7d 6d 3s 7h] [5d]  Agt 3.FKWBot/EssentialBot. bets $28.50  Agt 6.FKWBot/ReduxBot. folds  Uncalled bet of $28.50 returned to Agt 3.FKWBot/EssentialBot.  Agt 3.FKWBot/EssentialBot. wins the pot ($27.50)  \*\*\* SUMMARY \*\*\*  Seat 3: Agt 3.FKWBot/EssentialBot. collected ($27.50) |

A amostra a seguir, semelhantemente, mostra uma circunstância bastante frequente, quando o ReduxBot paga uma aposta no pré-flop e, após não obter uma mão forte no flop, desiste. Isso permite que o EssentialBot possa inferir apostas sem cartas privadas positivas e vença a mão tranquilamente.

|  |
| --- |
| Game #81901603248363: Table OpenTestBed - $0.50/$1.00  Seat 1: Agt 6.FKWBot/ReduxBot. ($100.00)  Seat 2: Agt 5.FKWBot/EssentialBot. ($100.00)  Seat 3: Agt 4.FKWBot/ReduxBot. ($100.00)  Seat 4: Agt 3.FKWBot/EssentialBot. ($100.00)  Seat 5: Agt 2.FKWBot/ReduxBot. ($100.00)  Seat 6: Agt 1.FKWBot/EssentialBot. ($100.00)  Agt 2.FKWBot/ReduxBot. posts the small blind of $0.50  Agt 1.FKWBot/EssentialBot. posts the big blind of $1.00  The button is in seat #4  \*\*\* HOLE CARDS \*\*\*  Dealt to Agt 6.FKWBot/ReduxBot. [6d Ts]  Dealt to Agt 5.FKWBot/EssentialBot. [5c 6c]  Dealt to Agt 4.FKWBot/ReduxBot. [Qs Kc]  Dealt to Agt 3.FKWBot/EssentialBot. [Ah 5s]  Dealt to Agt 2.FKWBot/ReduxBot. [4s 3c]  Dealt to Agt 1.FKWBot/EssentialBot. [7d Ad]  Agt 6.FKWBot/ReduxBot. folds  Agt 5.FKWBot/EssentialBot. raises to $3.50  Agt 4.FKWBot/ReduxBot. calls $3.50  Agt 3.FKWBot/EssentialBot. folds  Agt 2.FKWBot/ReduxBot. folds  Agt 1.FKWBot/EssentialBot. folds  \*\*\* FLOP \*\*\* [8h 4d 4c]  Agt 5.FKWBot/EssentialBot. bets $9.50  Agt 4.FKWBot/ReduxBot. folds  Uncalled bet of $9.50 returned to Agt 5.FKWBot/EssentialBot.  Agt 5.FKWBot/EssentialBot. wins the pot ($8.50)  \*\*\* SUMMARY \*\*\*  Seat 2: Agt 5.FKWBot/EssentialBot. collected ($8.50) |

Por fim, o próximo exemplo mostra uma situação onde o ReduxBot é finalmente agressivo e o EssentialBot possui o maior par da mesa. Nesse caso, contra outros oponentes, o melhor rumo pode ser nebuloso. Porém, contra um oponente cauteso, a opção de desistir é evidente. Outro fator relevante nesse e nos exemplos anteriores é a estratégia pré-flop do agente desenvolvido. Já que seu adversário participa de poucas mãos e, consequentemente, desiste constantemente ainda nessa rodada inaugural, é interessante ampliar o número de mãos que são aumentadas e já obter um lucro direto apenas pelas desistências do oponente. Como é possível perceber, o agente está com cartas privadas marginais em todos os históricos expostos, evidenciando esse comportamento.

Além disso, apesar das ações e dos provavéis motivos que levaram o agente a seguir um determinado rumo de jogo serem pormenorizados aqui, não há garantia que esse foi realmente o alicerce decisório por trás das decisões. Por isso, é preciso compreender que o agente não possui um processo racional explicítico e, ao contrário, apenas consulta informações probabilísticas e frequenciais para determinar a melhor jogada. Portanto, qualquer tentativa de descrever verbalmente o procedimento realizada é meramente hipotética, baseada nas premissas de funcionamento do EssentialBot.

|  |
| --- |
| Game #81901603172388: Table OpenTestBed - $0.50/$1.00  Seat 1: Agt 1.FKWBot/EssentialBot. ($100.00)  Seat 2: Agt 2.FKWBot/ReduxBot. ($100.00)  Seat 3: Agt 3.FKWBot/EssentialBot. ($100.00)  Seat 4: Agt 4.FKWBot/ReduxBot. ($100.00)  Seat 5: Agt 5.FKWBot/EssentialBot. ($100.00)  Seat 6: Agt 6.FKWBot/ReduxBot. ($100.00)  Agt 6.FKWBot/ReduxBot. posts the small blind of $0.50  Agt 1.FKWBot/EssentialBot. posts the big blind of $1.00  The button is in seat #5  \*\*\* HOLE CARDS \*\*\*  Dealt to Agt 1.FKWBot/EssentialBot. [Qh Td]  Dealt to Agt 2.FKWBot/ReduxBot. [8h 9d]  Dealt to Agt 3.FKWBot/EssentialBot. [7c 4c]  Dealt to Agt 4.FKWBot/ReduxBot. [Ah Qs]  Dealt to Agt 5.FKWBot/EssentialBot. [3h Ac]  Dealt to Agt 6.FKWBot/ReduxBot. [2s 5h]  Agt 2.FKWBot/ReduxBot. folds  Agt 3.FKWBot/EssentialBot. raises to $3.50  Agt 4.FKWBot/ReduxBot. calls $3.50  Agt 5.FKWBot/EssentialBot. folds  Agt 6.FKWBot/ReduxBot. folds  Agt 1.FKWBot/EssentialBot. folds  \*\*\* FLOP \*\*\* [Qc 3s 4d]  Agt 3.FKWBot/EssentialBot. bets $9.50  Agt 4.FKWBot/ReduxBot. calls $9.50  \*\*\* TURN \*\*\* [Qc 3s 4d] [5s]  Agt 3.FKWBot/EssentialBot. checks  Agt 4.FKWBot/ReduxBot. bets $28.50  Agt 3.FKWBot/EssentialBot. folds  Uncalled bet of $28.50 returned to Agt 4.FKWBot/ReduxBot.  Agt 4.FKWBot/ReduxBot. wins the pot ($27.50)  \*\*\* SUMMARY \*\*\*  Seat 4: Agt 4.FKWBot/ReduxBot. collected ($27.50) |

# ANEXO II – históricos contra agente previsível

As mãos contra o agente previsível não necessitam de uma descrição detalhada. Como todos os exemplos exploram a principal fraqueza do AlwaysCallBot, sua incapacidade de desistir, as jogadas demonstradas sempre visam auferir lucros através de apostas. Portanto, é sensato elucidar circunstâncias onde o agente construído possui mãos fracas e ainda assim infere apostas visando ser pago por mãos ainda piores. Por óbvio, essa decisão tem fundamentação em estrutura de dados complexas que comprovam a lucratividade dessa aposta.

|  |
| --- |
| Game #7045636224955: Table OpenTestBed  Seat 1: Agt 6.DemoBot/AlwaysCallBot. ($100.00)  Seat 2: Agt 5.FKWBot/EssentialBot. ($100.00)  Seat 3: Agt 4.FKWBot/EssentialBot. ($100.00)  Seat 4: Agt 3.FKWBot/EssentialBot. ($100.00)  Seat 5: Agt 2.FKWBot/EssentialBot. ($100.00)  Seat 6: Agt 1.FKWBot/EssentialBot. ($100.00)  Agt 6.DemoBot/AlwaysCallBot. posts the small blind of $0.50  Agt 5.FKWBot/EssentialBot. posts the big blind of $1.00  The button is in seat #6  \*\*\* HOLE CARDS \*\*\*  Dealt to Agt 6.DemoBot/AlwaysCallBot. [6d Kd]  Dealt to Agt 5.FKWBot/EssentialBot. [3c 7s]  Dealt to Agt 4.FKWBot/EssentialBot. [Ac 7d]  Dealt to Agt 3.FKWBot/EssentialBot. [Jd 9d]  Dealt to Agt 2.FKWBot/EssentialBot. [4h 2s]  Dealt to Agt 1.FKWBot/EssentialBot. [4c 7c]  Agt 4.FKWBot/EssentialBot. raises to $3.50  Agt 3.FKWBot/EssentialBot. folds  Agt 2.FKWBot/EssentialBot. folds  Agt 1.FKWBot/EssentialBot. folds  Agt 6.DemoBot/AlwaysCallBot. calls $3.00  Agt 5.FKWBot/EssentialBot. folds  \*\*\* FLOP \*\*\* [4d 4s 5s]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 4.FKWBot/EssentialBot. bets $9.00  Agt 6.DemoBot/AlwaysCallBot. calls $9.00  \*\*\* TURN \*\*\* [4d 4s 5s] [9h]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 4.FKWBot/EssentialBot. bets $27.00  Agt 6.DemoBot/AlwaysCallBot. calls $27.00  \*\*\* RIVER \*\*\* [4d 4s 5s 9h] [9s]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 4.FKWBot/EssentialBot. bets $60.50  Agt 6.DemoBot/AlwaysCallBot. calls $60.50  Agt 4.FKWBot/EssentialBot. shows [Ac 7d]  Agt 6.DemoBot/AlwaysCallBot. shows [6d Kd]  Agt 4.FKWBot/EssentialBot. wins the pot ($201.00)  \*\*\* SUMMARY \*\*\*  Seat 3: Agt 4.FKWBot/EssentialBot. showed [Ac 7d] and won ($201.00) |

|  |
| --- |
| Game #7045636224913: Table OpenTestBed  Seat 1: Agt 6.DemoBot/AlwaysCallBot. ($100.00)  Seat 2: Agt 5.FKWBot/EssentialBot. ($100.00)  Seat 3: Agt 4.FKWBot/EssentialBot. ($100.00)  Seat 4: Agt 3.FKWBot/EssentialBot. ($100.00)  Seat 5: Agt 2.FKWBot/EssentialBot. ($100.00)  Seat 6: Agt 1.FKWBot/EssentialBot. ($100.00)  Agt 6.DemoBot/AlwaysCallBot. posts the small blind of $0.50  Agt 5.FKWBot/EssentialBot. posts the big blind of $1.00  The button is in seat #6  \*\*\* HOLE CARDS \*\*\*  Dealt to Agt 6.DemoBot/AlwaysCallBot. [2s 8c]  Dealt to Agt 5.FKWBot/EssentialBot. [5h Ad]  Dealt to Agt 4.FKWBot/EssentialBot. [4h 4c]  Dealt to Agt 3.FKWBot/EssentialBot. [Jd 3d]  Dealt to Agt 2.FKWBot/EssentialBot. [3s Td]  Dealt to Agt 1.FKWBot/EssentialBot. [As 6d]  Agt 4.FKWBot/EssentialBot. raises to $3.50  Agt 3.FKWBot/EssentialBot. folds  Agt 2.FKWBot/EssentialBot. folds  Agt 1.FKWBot/EssentialBot. folds  Agt 6.DemoBot/AlwaysCallBot. calls $3.00  Agt 5.FKWBot/EssentialBot. folds  \*\*\* FLOP \*\*\* [Kd 7s Jc]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 4.FKWBot/EssentialBot. bets $9.00  Agt 6.DemoBot/AlwaysCallBot. calls $9.00  \*\*\* TURN \*\*\* [Kd 7s Jc] [6c]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 4.FKWBot/EssentialBot. bets $27.00  Agt 6.DemoBot/AlwaysCallBot. calls $27.00  \*\*\* RIVER \*\*\* [Kd 7s Jc 6c] [7h]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 4.FKWBot/EssentialBot. bets $60.50  Agt 6.DemoBot/AlwaysCallBot. calls $60.50  Agt 4.FKWBot/EssentialBot. shows [4h 4c]  Agt 6.DemoBot/AlwaysCallBot. shows [2s 8c]  Agt 4.FKWBot/EssentialBot. wins the pot ($201.00)  \*\*\* SUMMARY \*\*\*  Seat 3: Agt 4.FKWBot/EssentialBot. showed [4h 4c] and won ($201.00) |

|  |
| --- |
| Poker Game #7045636224888: Table OpenTestBed  1: Agt 6.DemoBot/AlwaysCallBot. ($100.00)  Seat 2: Agt 5.FKWBot/EssentialBot. ($100.00)  Seat 3: Agt 4.FKWBot/EssentialBot. ($100.00)  Seat 4: Agt 3.FKWBot/EssentialBot. ($100.00)  Seat 5: Agt 2.FKWBot/EssentialBot. ($100.00)  Seat 6: Agt 1.FKWBot/EssentialBot. ($100.00)  Agt 1.FKWBot/EssentialBot. posts the small blind of $0.50  Agt 6.DemoBot/AlwaysCallBot. posts the big blind of $1.00  The button is in seat #5  \*\*\* HOLE CARDS \*\*\*  Dealt to Agt 6.DemoBot/AlwaysCallBot. [7h Ad]  Dealt to Agt 5.FKWBot/EssentialBot. [4s 7s]  Dealt to Agt 4.FKWBot/EssentialBot. [5h Jd]  Dealt to Agt 3.FKWBot/EssentialBot. [3s 3h]  Dealt to Agt 2.FKWBot/EssentialBot. [Ah Jc]  Dealt to Agt 1.FKWBot/EssentialBot. [Ac Td]  Agt 5.FKWBot/EssentialBot. raises to $3.50  Agt 4.FKWBot/EssentialBot. folds  Agt 3.FKWBot/EssentialBot. calls $3.50  Agt 2.FKWBot/EssentialBot. folds  Agt 1.FKWBot/EssentialBot. folds  Agt 6.DemoBot/AlwaysCallBot. calls $2.50  \*\*\* FLOP \*\*\* [8d 6s 8c]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 5.FKWBot/EssentialBot. checks  Agt 3.FKWBot/EssentialBot. bets $12.00  Agt 6.DemoBot/AlwaysCallBot. calls $12.00  Agt 5.FKWBot/EssentialBot. folds  \*\*\* TURN \*\*\* [8d 6s 8c] [Ts]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 3.FKWBot/EssentialBot. bets $36.00  Agt 6.DemoBot/AlwaysCallBot. calls $36.00  \*\*\* RIVER \*\*\* [8d 6s 8c Ts] [Qs]  Agt 6.DemoBot/AlwaysCallBot. checks  Agt 3.FKWBot/EssentialBot. bets $48.50  Agt 6.DemoBot/AlwaysCallBot. calls $48.50  Agt 3.FKWBot/EssentialBot. shows [3s 3h]  Agt 6.DemoBot/AlwaysCallBot. shows [7h Ad]  Agt 3.FKWBot/EssentialBot. wins the pot ($204.00)  \*\*\* SUMMARY \*\*\*  Seat 4: Agt 3.FKWBot/EssentialBot. showed [3s 3h] and won ($204.00) |

|  |
| --- |
| Game #7045636224886: Table OpenTestBed  Seat 1: Agt 6.DemoBot/AlwaysCallBot. ($100.00)  Seat 2: Agt 5.FKWBot/EssentialBot. ($100.00)  Seat 3: Agt 4.FKWBot/EssentialBot. ($100.00)  Seat 4: Agt 3.FKWBot/EssentialBot. ($100.00)  Seat 5: Agt 2.FKWBot/EssentialBot. ($100.00)  Seat 6: Agt 1.FKWBot/EssentialBot. ($100.00)  Agt 3.FKWBot/EssentialBot. posts the small blind of $0.50  Agt 2.FKWBot/EssentialBot. posts the big blind of $1.00  The button is in seat #3  \*\*\* HOLE CARDS \*\*\*  Dealt to Agt 6.DemoBot/AlwaysCallBot. [6c 3h]  Dealt to Agt 5.FKWBot/EssentialBot. [4d Qs]  Dealt to Agt 4.FKWBot/EssentialBot. [5s Jc]  Dealt to Agt 3.FKWBot/EssentialBot. [Td 3s]  Dealt to Agt 2.FKWBot/EssentialBot. [Jd Kc]  Dealt to Agt 1.FKWBot/EssentialBot. [2d Ks]  Agt 1.FKWBot/EssentialBot. raises to $3.50  Agt 6.DemoBot/AlwaysCallBot. calls $3.50  Agt 5.FKWBot/EssentialBot. folds  Agt 4.FKWBot/EssentialBot. folds  Agt 3.FKWBot/EssentialBot. folds  Agt 2.FKWBot/EssentialBot. folds  \*\*\* FLOP \*\*\* [7d 8s 2c]  Agt 1.FKWBot/EssentialBot. bets $9.50  Agt 6.DemoBot/AlwaysCallBot. calls $9.50  \*\*\* TURN \*\*\* [7d 8s 2c] [7s]  Agt 1.FKWBot/EssentialBot. bets $28.50  Agt 6.DemoBot/AlwaysCallBot. calls $28.50  \*\*\* RIVER \*\*\* [7d 8s 2c 7s] [As]  Agt 1.FKWBot/EssentialBot. bets $58.50  Agt 6.DemoBot/AlwaysCallBot. calls $58.50  Agt 1.FKWBot/EssentialBot. shows [2d Ks]  Agt 6.DemoBot/AlwaysCallBot. shows [6c 3h]  Agt 1.FKWBot/EssentialBot. wins the pot ($201.50)  \*\*\* SUMMARY \*\*\*  Seat 6: Agt 1.FKWBot/EssentialBot. showed [2d Ks] and won ($201.50) |

GLOSSÁRIO

Esse glossário contém a definição informal dos termos e jargões do pôquer presentes nesse trabalho. Grande parte do conteúdo aqui presente foi retirado e traduzido de Sklansky (1994).

*All-in:* Jogador com todo seu dinheiro no *pot.*

*Ante:* Aposta obrigatória requerida por todos os jogadores antes do início de uma mão.

*Apostar:* ver *Bet.*

*Aumentar:* ver *Raise.*

*Bankroll:* Quantidade total de dinheiro disponível para apostas, além do *stack.*

*Bet:* primeiro jogador a por dinheiro voluntariamente no *pot* em uma determinada rodada*.*

*Blefar:* apostar com uma mão com pouco ou nenhum valor visando à desistência do oponente.

*Blind:* Apostas forçadas presente na primeira rodada do Texas Hold’em, divididas em *Small Blind* e *Big Blind*. Ao fim de cada mão, as *blinds* rotacionam no sentido horário.

*Board:* As cartas comunitárias.

*Call:* Igualar a aposta ou *raise* de outro jogador.

*Check:* Não apostar qualquer valor, passando para o próximo jogador.

*Expected Value:* o ganho (ou prejuízo) médio de uma aposta no longo prazo.

*Flop:* segunda rodada do jogo, onde as três primeiras cartas comunitárias são reveladas.

*Flush:* cinco cartas do mesmo naipe.

*Fold:* desistir da mão ao negar pagar qualquer aposta.

*Heads-up:* jogo com apenas dois jogadores; mano-a-mano.

*Mão:* as cartas privadas de um jogador ou o valor das cinco melhores cartas de um jogador.

*Mesa:* quando referente às cartas, ver *Board.* Caso contrário, ver *Check.*

*Overpair:* par maior que a maior carta presente na mesa comunitária.

*Posição dianteira:* Lugar relativo na mesa cujo jogador age antes da maioria dos jogadores.

*Posição intermediária:* Lugar relativo na mesa, entre as posições dianteiras e traseiras.

*Posição traseira:* Lugar relativo na mesa cujo jogador age depois da maioria dos jogadores.

*Pot:* quantidade total de apostas em disputa pelos jogadores em uma mão.

*Raise:* Apostar um valor superior ao atual; aumentar a aposta de um adversário.

*River:* quarta e última rodada de apostas, onde a quinta carta comunitária é revelada.

*Turn:* terceira e última rodada de apostas, onde a quarta carta comunitária é revelada.

*Stack:* quantia que um dado jogador possui na mesa.

*Sequência:* ver *Straight.*

*Showdown:* fim da mão, quando os participantes mostram suas cartas privadas.

*Straight:* cinco cartas de naipes distintos com valores sequenciais.

*Valor esperado:* ver *Expected value.*