UNIVERSIDADE FEDERAL DO PARÁ
NÚCLEO DE DESENVOLVIMENTO AMAZÔNICO EM ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA

ADRIANO AUGUSTO ADDARIO DOS SANTOS

UM PROTÓTIPO DE SOFTWARE PARA SIMULAÇÃO DE SISTEMAS MULTIAGENTES BASEADOS NA ABORDAGEM DE RUSSELL/NORVIG E NA TEORIA DOS JOGOS

Tucuruí/Pará

2019

ADRIANO AUGUSTO ADDARIO DOS SANTOS

UM PROTÓTIPO DE SOFTWARE PARA SIMULAÇÃO DE SISTEMAS MULTIAGENTES BASEADOS NA ABORDAGEM DE RUSSELL/NORVIG E NA TEORIA DOS JOGOS

Dissertação apresentada ao Programa de Pós-Graduação em Computação Aplicada do Núcleo de Desenvolvimento Amazônico em Engenharia, da Universidade Federal do Pará, como requisito para a obtenção do título de Mestre em Computação Aplicada.

Orientador: Otávio Noura Teixeira

Tucuruí/Pará

2019

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD Sistema de Bibliotecas da Universidade Federal do Pará

Gerada automaticamente pelo módulo Ficat, mediante os dados fornecidos pelo(a) autor(a)

S237p Santos, Adriano Augusto Addario dos

Um protótipo de software para simulação de sistemas multiagentes baseados na abordagem de Russell/Norvig e na teoria dos jogos / Adriano Augusto Addario dos Santos. — 2019.

107 f. : il. color.

Orientador(a): Prof. Dr. Otávio Noura Teixeira Dissertação (Mestrado) - 1, , Universidade Federal do Pará, Tucuruí, 2019.

1. Agentes inteligentes. 2. Teoria dos jogos. 3. Torneios computacionais. 4. Estratégias de comportamento. I. Título.

CDD 006.3

ADRIANO AUGUSTO ADDARIO DOS SANTOS

UM PROTÓTIPO DE SOFTWARE PARA SIMULAÇÃO DE SISTEMAS MULTIAGENTES BASEADOS NA ABORDAGEM DE RUSSELL/NORVIG E NA TEORIA DOS JOGOS

Dissertação apresentada ao Programa de Pós-Graduação em Computação Aplicada do Núcleo de Desenvolvimento Amazônico em Engenharia, da Universidade Federal do Pará, como requisito para a obtenção do título de Mestre em Computação Aplicada.

Orientador: Otávio Noura Teixeira

Aprovada em 30 de maio de 2019.

BANCA EXAMINADORA:

Doutor / Otávio Noura Teixeira / Universidade Federal do Pará – Orientador

Doutor / Rafael Suzuki Bayma / Universidade Federal do Pará

Doutor / Gustavo Augusto Lima de Campos / Universidade Estadual do Ceará

Tucuruí/Pará

2019

AGRADECIMENTOS

Agradeço primeiramente a Deus, pois Ele me concedeu força para executar esse trabalho e sem Ele nada teria sido feito.

Aos meus pais **Fernando** e **Maria José Santos**, por todo apoio, carinho e dedicação. Meu pai por todo seu sacrifício, suporte financeiro e acima de tudo sua amizade e companheirismo. Minha mãe por suas incansáveis orações, conselhos e muito carinho.

À **Universidade Federal do Pará**, ao **Programa de Pós-Graduação em Computação Aplicada (PPCA)**, e em especial ao meu amigo e orientador **Otávio Noura Teixeira (tio Noura)**, pela parceria, amizade, e gigantesca paciência. Muito obrigado por tudo!

Minha irmã, Bianca Addario, por mesmo estando longe sempre aparecer nos momentos certos e me proporcionar uma imensa injeção de ânimo, que me ajudou principalmente nas noites em claro.

Minha namorada e companheira de equipe, Natalia Araújo, por todo carinho, dedicação e cuidado que teve ao participar no processo de desenvolvimento desta dissertação.

E por fim, mas não menos importante, ao aluno de Engenharia de Computação Renuá Almeida, que entrou na equipe de desenvolvimento próximo do final e ajudou a dar uma dimensão ainda maior para o simulador.

"Um herói pode ser qualquer um,
até mesmo um homem fazendo algo tão
simples e reconfortante como colocar um
casaco em torno dos ombros de um
menino, para deixá-lo saber que o mundo
não tinha terminado."
Batman – O Cavaleiro das Trevas

RESUMO

A base para o desenvolvimento da inteligência artificial tem em seu alicerce um conceito primordial – os agentes inteligentes. Estudar, compreender e melhorar o comportamento dessas entidades autônomas são algumas das motivações deste trabalho. Compreender e melhorar o comportamento dessas entidades autônomas são algumas das motivações deste trabalho, o qual busca apresentar um protótipo de software, que viabiliza o estudo dos comportamentos de agentes, utilizando a estrutura proposta por Russell e Norvig e introduzindo uma forma de classificação das estratégias de comportamentos utilizadas em jogos do Dilema do Prisioneiro. É utilizado como base, os torneios computacionais realizados por Axelrod em 1984, assim como outros torneios que aconteceram nesses mais de 30 anos de pesquisas voltadas ao Dilema do Prisioneiro Iterado. Como parte integrante desse trabalho, foi utilizado a biblioteca de Vincent Knight, que conta 242 estratégias de comportamento implementadas em Python. Somado a biblioteca, foram adicionas mais 512 estratégias, formando assim uma base de 754 estratégias todas implementadas em Python. Ao unir o campo da inteligência artificial com a teoria dos jogos, o intuito é utilizar o melhor das duas áreas para aplicação em solução de problemas de alta complexidade.

Palavras-chave: Agentes inteligentes. Teoria dos jogos. Torneios computacionais. Estratégias de comportamento.

ABSTRACT

The basis for the development of artificial intelligence is based on a primordial concept - intelligent agents. Studying, understanding and improving the behavior of these autonomous entities are some of the motivations of this work. Understanding and improving the behavior of these autonomous entities are some of the motivations of this work, which seeks to present a software prototype, which enables the study of agent behaviors, using the structure proposed by Russell and Norvig and introducing a way of classifying the strategies of behaviors used in Prisoner's Dilemma games. It is used as a basis for Axelrod's computational tournaments in 1984, as well as other tournaments that took place in these more than 30 years of research into the Iterated Prisoner's Dilemma. As an integral part of this work, we used Vincent Knight's library, which features 242 behavior strategies implemented in Python. In addition to the library, 512 more strategies were added, thus forming a base of 754 strategies all implemented in Python. By uniting the field of artificial intelligence with game theory, the aim is to use the best of both areas for application in highly complex problem solving.

Keywords: Intelligent agents. Theory of games. Computer Tournaments. Behavior strategies.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação e Justificativa

Durante muitos anos, a Inteligência Artificial não passou de plano de fundo para filmes de ficção científica. A ideia de uma sociedade povoada por robôs inteligentes que, mais do que realizar tarefas, interagem de maneira totalmente humana serve de base para várias produções de sucesso. Mas com o passar dos anos a tecnologia evoluiu e começou a fazer parte do cotidiano. Coisas como dirigir, marcar compromissos, por exemplo, consideradas ações exclusivamente humanas foram modeladas e hoje já são realizadas por sistemas inteligentes.

A Inteligência Artificial é um importante campo de estudo que teve início logo após a segunda guerra mundial e que atualmente contempla uma diversa variedade de subcampos. Seu objetivo principal é o desenvolvimento de sistemas que simulem a capacidade humana de raciocínio, percepção e tomada de decisão visando à resolução de problemas inerentes das mais diversas áreas (DAMIÃO; CAÇADOR; LIMA, 2014, p. 02).

Em busca de maior diversidade de estratégias para desenvolvimento de uma Inteligência Artificial mais próxima da capacidade humana, diversas técnicas foram empregadas para tal propósito. Focando nessa diversidade Teixeira (2015) destaca que uma área específica da matemática, denominada Teoria dos Jogos, tem despertado interesse da ciência da computação por sua versatilidade e vem sendo utilizada em avanços relevantes na área de Inteligência Artificial e cibernética.

Compreender os comportamentos considerados ótimos em determinadas situações se torna um pré-requisito indispensável para a modelagem de Agentes Inteligentes. Russell e Norvig (2013), destacam que o conceito de agente inteligente é o ponto central de qualquer abordagem de inteligência artificial.

Define-se Agentes Inteligentes como entidades autónomas, dotadas de uma base de conhecimento e capazes de interagir com o meio em que estão tomando assim decisões que irão auxiliar ou até mesmo substituir o trabalho de um humano (RUSSELL E NORVIG, 2013).

Pode-se citar como exemplo de trabalho conjunto entre a teoria dos jogos e agentes inteligentes, o famoso problema do Dilema do Prisioneiro. De forma resumida, esse problema define uma situação de conflito de interesses, onde dois indivíduos são presos e colocados em celas diferentes e sem comunicação entre eles (POUNDSTONE, 1992).

Vislumbrando as inúmeras possibilidades da utilização da teoria dos jogos, atrelada aos conceitos fundamentais que norteiam os estudos na área de inteligência artificial. Esse trabalho busca integrar a construção de agentes (baseados em Russell e Norvig, 2013) e os comportamentos resultantes dos torneios computacionais de Axelrod (1984) em um software de fácil utilização.

1.2 Objetivos

1.2.1 Geral

Desenvolver um protótipo de software para simulação e estudo dos torneios computacionais e estratégias de comportamento.

1.2.2 Específicos

Criar uma síntese das estruturas de agentes apresentadas por Russell e Norvig (2013).

Classificar as estratégias de comportamento presentes nos torneios computacionais de Axelrod usando como base os programas de agentes apresentados por Russell e Norvig (2013).

Ressimular os principais torneios computacionais sobre o Dilema do Prisioneiro Iterado presentes na literatura.

1.3 Contribuições da Dissertação

As contribuições advindas do desenvolvimento deste trabalho são as seguintes:

1. Disponibilizar uma ferramenta para auxílio no estudo da teoria dos jogos, que seja facilmente utilizado por qualquer pesquisador, de qualquer área de pesquisa;

2. Concepção e classificação da visualização das estratégias de comportamento baseados nas estruturaras de agentes;
3. Enriquecer a biblioteca Axelrod, desenvolvida por Vincent Knight, através do acréscimo de estratégias que não haviam sido implementadas.

1.4 Estrutura Da Dissertação

Este trabalho está estruturado em **cinco capítulos** e **três anexos**. No capítulos um é feita a apresentação e a as considerações iniciais deste trabalho, o Capítulo dois apresenta a fundamentação teórica, baseada em duas vertentes. A primeira consiste na descrição e principais conceitos sobre agentes inteligentes, suas estruturas e caracterização de ambiente. A segunda é focada no conceito de Teoria dos Jogos e suas definições, assim como o dilema do prisioneiro e os torneios computacionais originados dessa exemplificação da Teoria dos Jogos.

No Capítulo três é feito o detalhamento do protótipo, além da definição e estrutura do modelo de classificação baseado nas arquiteturas de agentes. O Capítulo quatro foi reservado para avaliação do protótipo e validação através da reprodução dos torneios de Axelrod (1984), Borges (1996), Knight (2017). E, por fim, no Capítulo cinco são apresentadas as considerações finais, através das conclusões, limitações da dissertação e os trabalhos futuros.

No anexo, são apresentados materiais que foram considerados essências e precisam estar em um local de fácil consulta. No anexo A, estão todas as descrições das 14 estratégias apresentadas no primeiro torneio realizado por Axelrod, e o anexo B contém as 64 estratégias que compõem o segundo torneio de Axelrod. No anexo C, está o resultado da simulação do torneio de Vincent Knight.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Agentes Inteligentes

2.1.1 Apresentação

O desejo por máquinas e/ou entidades que resolvam problemas não é recente. Por volta de 800 a.C, já existiam relatos do desejo do homem de possuir máquinas para auxiliar em suas atividades cotidianas. No canto XVIII de Ilíadas, Homero apresenta as *trípodes,* objeto "automatizado" por Hefesto, deus Grego do fogo:

> "para ficarem de pé em volta do muro da sua casa bem construída; e rodas de ouro colocara sob a base de cada uma, para que entrassem, automáticas, na reunião divina e de novo voltassem a casa, maravilha de se ver! Neste estado de aperfeiçoamento estavam já; faltava pôr-lhes as orelhas trabalhadas; mas ele preparava-as e fabricava os rebites.*"* (HOMERO, 2005 p. 370-375).

Apesar de ser apenas uma ficção, esse fascínio por máquinas que realizem atividades de forma autônoma esteve presente em diversas literaturas ao longo dos séculos. Em seu trabalho Nakamiti (2009), faz referência a grandes obras da literatura e inúmeros fatos históricos. Fatos estes que vão desde Thomas Hobbes, em sua publicação "LEVIATHAN", onde ressalta que os homens podem criar uma inteligência. Passando por George Moore (1893) e seu "homem a vapor", chegando até Isaac Asimov (1942) com seu conto "Runaround", que apresentou o termo "robótica".

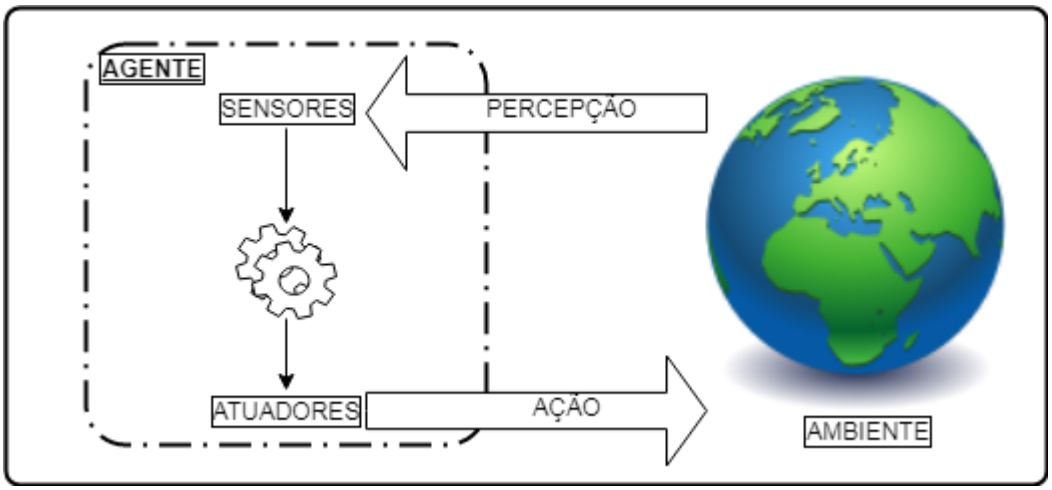
Todos as situações apresentadas convergem para uma entidade que tenha a capacidade de interagir com o ambiente que o rodeia. Essa entidade, atualmente, recebe o nome de agente.

Freitas (2002) destaca que como em muitas áreas da computação, ainda não há um consenso entre as diferentes definições de agentes, abrindo um amplo espectro para discussão.

Mas na literatura é comum a definição formal de agentes convergir para sistemas intencionais cujos objetivos são observar o ambiente por meio de um ou mais sensores, processar as informações e modificar o ambiente utilizando seus atuadores (RUSSELL E NORVIG 2013, REZENDE 2005, BRENNER ET. AL, 1998; MÜLLER, 1996).

De uma maneira mais ilustrativa, usando como base o trabalho de Russell e Norvig (2013) podemos visualizar os principais elementos da interação de um agente com o ambiente (figura 1).

Figura 1. Interação agente-ambiente



FONTE: Autor (baseado em Russell e Norvig (2013))

Como por definição, um agente nada mais é do que um conjunto de entradas, processamento e saídas. Teixeira (2012) afirma que é possível retratar matematicamente um agente através da seguinte função:

$$f : P^* \rightarrow A \qquad (2.1)$$

Onde:

f : *é a função do agente;*
P : *é conjunto de percepções do agente;*
A : *é o conjunto de ações do agente.*

Tomando como base o enunciado matemático acima, define-se como programa do agente a implementação computacional de f ,tendo como forma básica a estrutura a seguir figura 2. (RUSSELL E NORVIG, 2013).

Figura 2. Pseudo código de um agente

```
1  função_agente(percepcoes):
2      programa
3  retorna (acao);
```

FONTE: Russell e Norvig (2013)

Apesar de parecer simples, esse algoritmo é a base da construção da interação de um agente com o ambiente (Russell e Norvig, 2013).

2.1.2 Agente Racional

O conceito de agente racional é um conceito central para a inteligência artificial, pois um agente racional busca em suas ações sempre um estado de “fazer tudo certo”. Para entender esse conceito, é fundamental compreender o que seria “fazer tudo certo” (RIVERO, 1999).

Existem diferentes maneiras de definir esse estado, será utilizada a definição de Russell e Norvig (2013), que apresentam “que a ação certa é aquela que fará com que o agente obtenha o maior sucesso”. Para que seja avaliado o sucesso de um agente, faz-se necessário uma forma de medir esse desempenho, para esse propósito é necessário o estabelecimento de uma **medida de desempenho.**

A medida de desempenho se trata de um critério que serve de parâmetro para avaliar o agente, é importante destacar que não existe uma fórmula fixa para todos os agentes. Essa medida será definida, baseada em que objetivos o agente precisa alcançar.

2.1.3 Estrutura De Agentes

Um agente é constituído de um programa somado as suas estruturas de sensores e atuadores. A esse conjunto é dado o nome de arquitetura do agente.

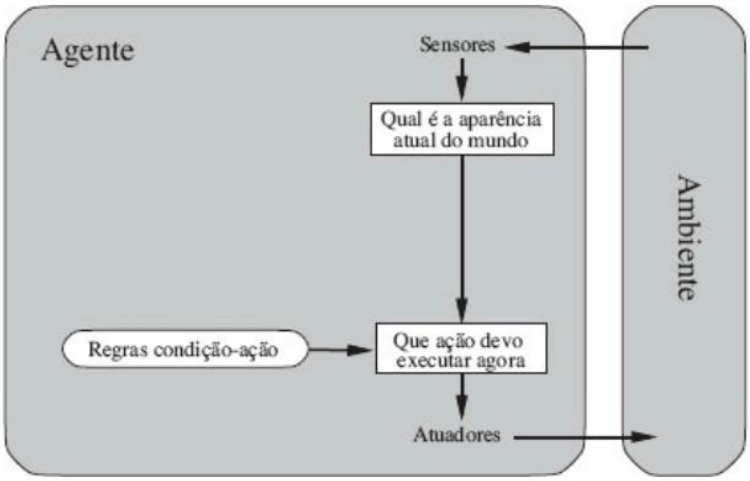
Será utilizado como base as arquiteturas utilizadas no segundo capítulo do trabalho de Rossell e Norvig (2013), onde são destacadas as seguintes arquiteturas:

- Agente reativo simples;
- Agente reativo baseado em modelo;
- Agente baseado em objetivo;
- Agente baseado na utilidade.

2.1.3.1 Agente Reativo Simples (ARS)

Trata-se do agente mais básico, ele não possui uma memória, então suas ações são baseadas apenas em sua percepção atual do ambiente. Sua estrutura genérica é apresentada em Russell e Norvig (2013, p.47).

Figura 3. Arquitetura de um agente reativo simples



FONTE: Russell e Norvig (2013)

Em um agente reativo simples, existe uma estrutura de controle do tipo *se-então* com as regras que o agente deve seguir caso seus sensores recebam alguma informação contida em suas regras. É importante ressaltar que cada nova decisão não leva em consideração as ações executadas anteriormente e nem os estados anteriores do ambiente.

O algoritmo proveniente da estrutura mostrada na figura 3 é uma função que recebe percepções e retorna uma ação.

Figura 4.Algoritmo de um agente reativo simples

```
função AGENTE-REATIVO-SIMPLES (percepção) retorna uma ação
   variáveis estáticas: regras, um conjunto de regras condição-ação

   estado ← INTERPRETAR-ENTRADA (percepção)
   regra ← REGRA-CORRESPONDENTE (estado, regras)
   ação ← AÇÃO-DA-REGRA [regra]
   retornar ação
```

FONTE:Russell e Norvig (2013)

Nesse programa a variável *estado* é alimentada por uma função INTERPRETAR-ENTRADA que cria um modelo abstrato do estado atual de percepção. A função REGRA-CORRESPONDENTE utiliza a variável estado e retorna a primeira regra correspondente a descrição do estado dado utilizando um conjunto de regras pré-estabelecidas. Por fim, a função AÇÃO-DA-REGRA que alimenta a variável *ação*, para ser executada pelos atuadores.

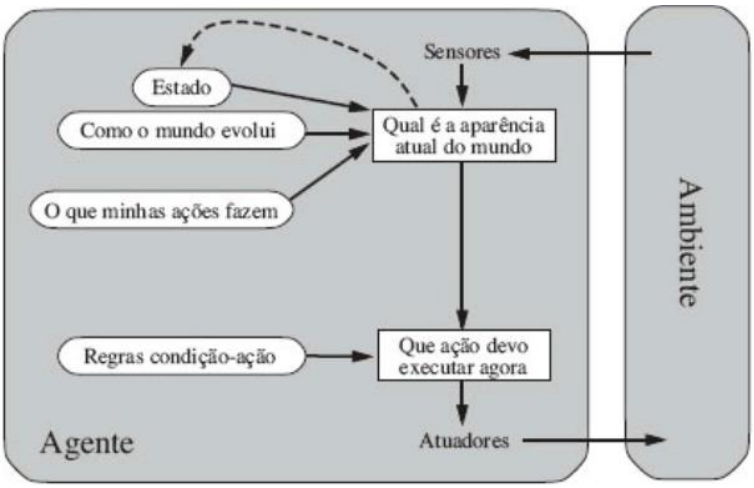
Por se tratar de um modelo simples, eventualmente um ARS cair em laços, para solucionar esse problema o agente pode utilizar-se da aleatoriedade para tomar suas decisões. Outro ponto é que um ARS fica limitado a aplicações onde as decisões possam ser tomadas levando em conta apenas o estado atual do ambiente.

2.1.3.2 Agente Reativo Baseado em Modelo (ARBM)

Um ARBM possui todas as características de um ARS, o que o torna diferente é a necessidade de guardar um histórico das iterações já realizadas, e em seu estado interno ele deve refletir sobre alguns aspectos não observáveis do estado atual de interação.

Ao analisarmos a estrutura genérica de um ARBM figura 4, pode-se notar a preocupação do agente com a evolução do ambiente e o que suas ações alteram no mesmo.

Figura 5.Agente baseado em modelo



FONTE: Russell e Norvig (2013, p.49)

O algoritmo proveniente dessa estrutura, além de utilizar as funções presentes no algoritmo do ARS, utiliza uma implementação importante de atualização do estado de mundo.

Figura 6.Algoritmo de um agente reativo baseado em modelo

função AGENTE-REATIVO-BASEADO-EM-MODELOS (*percepção*) **retorna** uma ação
  persistente: *estado*, a concepção do agente do estado atual do mundo
  modelo, uma descrição de como o próximo estado depende do estado atual e da ação
  regras, um conjunto de regras condição-ação
  ação, a ação mais recente, inicialmente nenhuma

  estado ← ATUALIZAR-ESTADO (*estado*, *ação*, *percepção*, *modelo*)
  regra ← REGRA-CORRESPONDENTE (*estado*, *regras*)
ação ← *regra*, *AÇÃO*
retornar *ação*

FONTE: Russell e Norvig (2013)

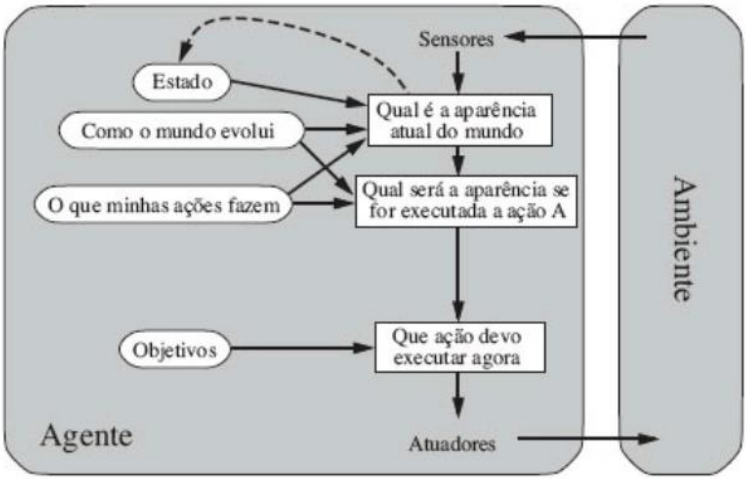
No caso do ARBM, o estado é atualizado por uma função ATUALIZAR-ESTADO, essa função é responsável por dar para o agente uma nova visão do mundo, levando em conta as mudanças ocorridas desde a última iteração, sejam elas motivadas pela ação do agente, ou não.

O agente nem sempre terá um modelo atualizado do mundo completamente fiel, geralmente o mais próximo que o agente terá é um “palpite”.

2.1.3.3 Agente Baseado em Objetivo (ARBO)

Nem sempre conhecer o estado atual ou como suas ações modificam o ambiente são suficientes para decidir qual decisão tomar. A estrutura do ARBO coloca o objetivo como novo fator que direciona as tomadas de decisão. A figura 6 mostra a estrutura de um agente baseado em objetivo.

Figura 7.Agente baseado em objetivo



FONTE: Russell e Norvig (2013)

Diferente de um ARS, o ARBO não trabalha baseado em uma função *CONDIÇÃO-AÇÃO direta,* como no ARBM suas ações são baseadas no que suas ações modificaram o mundo e como o mundo se parece agora, porém vai além quando utiliza o objetivo que apresenta a descrição da situação desejada.

Às vezes a seleção da ação baseada no objetivo é direta. Em outras ela passa por busca e planejamento, o que começa a tornar a modelagem desse agente um pouco mais complexa que os demais.

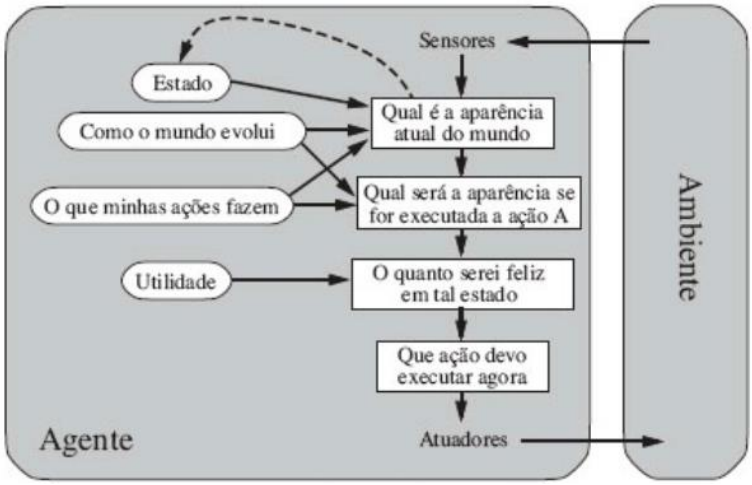
Com essas características, o ARBO pode parecer um tanto ineficiente, mas sua estrutura permite maior flexibilidade, uma vez que o agente pode utilizar seus conhecimentos para melhor se adaptar as condições impostas no objetivo.

Outro ponto interessante é que em um ARS, para utilizá-lo com outra finalidade seria necessário reescrever a maior parte (se não toda) de sua função CONDIÇÃO-AÇÃO, em um ARBO esse comportamento de mudança ficaria atrelado ao objetivo isso faria todos os comportamentos relevantes fossem alterados para atender as novas especificações do objetivo.

2.1.3.4 Agente Baseado em Utilidade (ARBU)

Em determinados casos, apenas o objetivo, não é suficiente para que o agente realize suas tarefas buscando o melhor desempenho. Por isso, um agente baseado em utilidade utiliza a definição de um ARBO, atrelado com uma função utilidade que busca os melhores resultados de suas ações.

Figura 8.Agente baseado em utilidade



FONTE: Russell e Norvig (2013

Uma função utilidade mapeia um estado e define um número real para definir um nível de “felicidade” para o agente. Essa felicidade nada mais é do que um valor considerado ideal, que o agente busca maximizar.

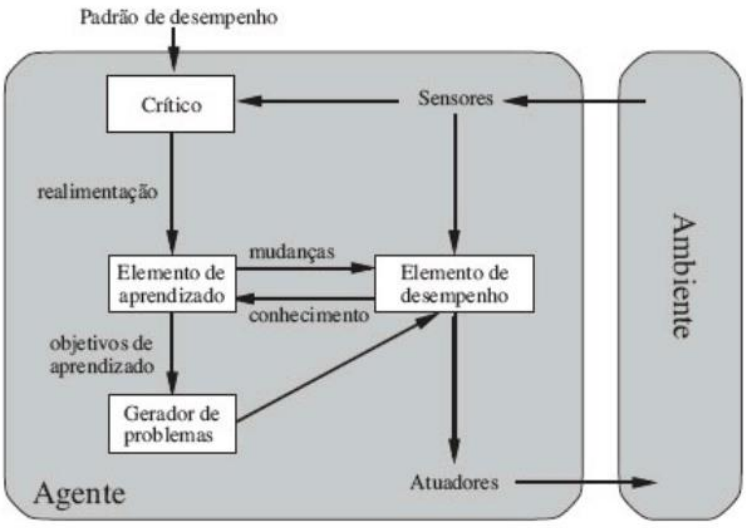
> Tecnicamente falando, um agente racional baseado em utilidade escolhe a ações que maximiza a utilidade esperada dos resultados da ação, isto é, a utilidade que o agente espera obter, em média, dadas as probabilidades e as utilidades de cada resultado. (RUSSELL E NORVIG 2013, p. 51)

2.1.3.5 Agente com Aprendizado (ACA)

O processo de construir máquinas com inteligência requer um método mais eficiente. Pensando nisso, Turing (1950, apud RUSSELL E NORVIG 2013, p.51) propõe método de construir máquinas com a capacidade de aprendizagem e ensiná-las.

O aprendizado tem outra vantagem, um agente pode operar em um ambiente inicialmente desconhecido e se tornar mais eficiente à medida que aprende como interagir com o ambiente. Na figura 8, temos a estrutura base de um agente com aprendizagem.

Figura 9.Agente baseado em utilidade



FONTE: Russell e Norvig (2013)

De maneira genética, um agente com aprendizado apresenta quatro elementos principais. Eles são:

- **Elemento de aprendizado:** responsável pela execução do aperfeiçoamento do agente;
- **Elemento de desempenho:** Fica responsável pela seleção das ações externas, ele recebe percepções e decide sobre as ações;
- **Crítico:** O crítico informa ao elemento de aprendizado como o agente está se comportando em relação a um padrão fixo de desempenho;
- **Gerador de problemas:** é responsável por sugerir novas experiências ao agente, buscando a expansão do aprendizado.

2.1.4 Ambiente: Definição E Características

Um dos pontos fundamentais para a escolha de um agente é entender em que tipo de ambiente ele estará inserido. Al-Zaniti et al. (2013) ressaltam que a definição do ambiente afeta diretamente a construção do agente.

Serão listadas as principais propriedades de um ambiente segundo Russell e Norvig (2013), Juchem e Bastos (2001) e Al-Zinati at al.(2013), assim como suas respectivas descrições.

- **Completamente observável** ou **parcialmente observável**

Um ambiente pode ser considerado completamente observável quando o agente possui informações total acesso aos estados do ambiente através de seus sensores. Esse tipo de ambiente é bastante conveniente, uma vez que o agente não precisa manter um estado interno para controla a aparência do mundo.

Por consequência, um ambiente parcialmente observável surge quando o agente não consegue obter todas as informações do ambiente (seja por ruído ou falta de sensores), nesse tipo de ambiente o agente precisa manter um estado interno que mapeia a situação do ambiente baseado em suas observações.

- **Determinístico** ou **Estocástico**

Se o próximo estado do ambiente é completamente determinado pelo estado atual e pela ação executada pelo agente, dizemos que o ambiente é determinístico. A princípio esse tipo de ambiente é bastante conveniente, pois o

agente não precisa lhe dar com incertezas no caso de um ambiente completamente observável e determinístico.
Quando o ambiente se comporta de maneira aleatória, esse tipo de ambiente é considerado estocástico.

- **Episódico** ou **sequencial**

Em um ambiente de tarefa episódico, a experiência do agente é dividida em episódios. Ou seja, o agente recebe uma percepção e executa uma ação, para que o ambiente seja considerado episódico é essencial que o próximo episódio não dependa das ações tomadas no episódio atual.
Por outro lado, em um ambiente sequencial, a decisão atual terá efeito sobre a sequência dos acontecimentos. Ambientes episódicos são muito mais simples que ambientes sequenciais porque o agente não precisa pensar à frente.

- **Estático** ou **dinâmico**

Caso o ambiente possa sofrer qualquer tipo de mudança enquanto um agente está atuando sobre ele, pode-se afirmar que esse tipo de ambiente é dinâmico, caso ele permaneça imutável é dito que o ambiente é do tipo estático.
Nos casos em que o ambiente muda de acordo com a atuação de um agente, esse tipo de ambiente e classificado com um ambiente semi-dinâmico.

- **Discreto** ou **contínuo**

Um ambiente tido com discreto apresenta um número finito ações e percepções que um agente pode receber e executar. Quando o ambiente fornece ao agente informações que não podem ser tratadas pontualmente, esse ambiente passa a ser considerado contínuo.

- **Monoagente** ou **multiagente**

Um ambiente de agente único é aquele onde apenas um agente atua sobre ele. Já em um ambiente multiagente, vários agentes interagem com o ambiente e entre si, gerando comportamentos de competição e/ou cooperação.

- **Competitivo** ou **cooperativo**

  Em um ambiente competitivo, o um agente busca maximizar seu desempenho e virtude da minimização do desempenho de outros agentes. Existem ambientes onde os agentes podem buscar maximizar seus desempenhos sem depender da diminuição do desempenho de outros agente, além de conseguirem através da cooperação maximizar ambos desempenhos, esse tipo de ambiente é tido como cooperativo.

Ao fazer uma análise sobre os ambientes reais, diversos autores (Al-Zinati et al. (2013), Howe et al. (2006), Steel et al. (2010), Kuiper e Wenkstern (2011), Luke et al. (2005), North et al. (2005), Amouroux et al. (2009)) ressaltam que a melhor caracterização é um ambiente do tipo parcialmente observável, estocástico, sequencial, dinâmico, contínuo, parcialmente cooperativo e multiagentes. Por retratar as diversas variáveis e inconsistências de um ambiente real.

2.1.5 Sistemas Multi-Agentes (MAS)

Sistemas multiagentes é uma área da Inteligência Artificial Distribuída que estuda o comportamento de agentes inteligentes, em relação a problemas em que a solução não pode ser obtida através de interações individuais Wooldridge e Jennings (1995).

Esses sistemas têm aplicações em diferentes áreas. Correa (2013) lista a grande variedade de aplicações com uma sequência de trabalhos correlatos na área.

> Como controle distribuído (Riedmiller et al., 2000; Stephan et al., 2000; Wiering, 2000; Bakker et al., 2005), times robóticos (Balch, 1997; Luke et al., 1997; Luke, 1998; Fernandez e Parker, 2001; Stone e Veloso, 2000; Hladek, 2007), controle de navegação (Bowling e Veloso, 2002; Hu e Wellman, 2003; Calvo e Romero, 2007), controle e planejamento de rotas (Boyan e Littman, 1993; Choi e Yeung, 1995; Tillotson et al., 2004), programação de elevadores (Crites e Barto, 1998), balanceamento de carga (Schaerf et al., 1995), negociação (trading) automática entre agentes (Wellman et al., 2003; Hsu e Soo, 2001; Lee e Oo, 2002; Oo et al., 2002; Tesauro e Kephart, 2002; Raju et al., 2003), precificação dinâmica de produtos de varejo (Raju et al. 2003), etc.

Essa área é vista por muitos pesquisadores como um horizonte cheio de novas perspectivas para a solução de problemas de alta complexidade. Pois os, sistemas multiagentes apresentam um conceito adaptativo que se aplica aos mais variados problemas de mundo real (AL-ZINATI ET AL. ,2013).

2.1.6 Ferramentas Para Construção De Sistemas Multi-Agentes

Existem várias ferramentas no mercado que buscam servir como base para a criação de sistemas multiagentes. No site do MASON os próprios criadores do software listam outros 09 softwares que tem a mesma finalidade.

Na revisão da literatura é recorrente o aparecimento de 4 ferramentas que são o MASON, REPAST, GAMA e DIVAS. Como ressaltado anteriormente, existe uma infinidade de softwares para criação de sistemas multiagentes. Foram escolhidos apenas estes pela frequência com que aparecem na literatura e por simularem ambientes com características reais citados na introdução desse tópico.

2.1.6.1 MASON

Desenvolvido em Java, o *MASON* é interessante desde a origem de sua sigla que do inglês (**M**ulti-**A**gent **S**imulator **O**f **N**eighborhoods... or **N**etworks... or something), em tradução livre para o português temos, simulador multi-agentes de proximidades ou de rede ou de algo assim.

Esse bom humor direciona a construção da ferramenta que busca ser a base para simulações maiores, utilizando a linguagem Java. Em suas características estão ser rápido, portátil e relativamente pequeno. A ferramenta conta com um manual de mais de 300 páginas, e uma extensa documentação online, além de ter uma lista de discussão organizada e atualizada.

Para agrupar todo esse conhecimento, a equipe de desenvolvimento, mantem uma página web(https://cs.gmu.edu/~eclab/projects/mason/#Features), com suas principais publicações e trabalhos realizados com o MASON, assim como também apresenta as principais extensões da ferramenta.

2.1.6.2 REPAST

O *Repast* é uma ferramenta *open source*, que vem sendo desenvolvida ao longo de mais de 15 anos de pesquisa. É uma plataforma de modelagem baseada em agentes e voltada para a simulação multiagentes.

Em sua última versão, possui dois componentes. O primeiro *Repast Simphony* (https://repast.github.io/repast_simphony.html), baseado em Java, busca ser uma ferramenta de para uso em atividades de baixa e média complexidade. Em sua documentação, existe um direcionamento para diferentes níveis de públicos. Começando pelo *ReLogo getting started*, que busca se valer de uma linguagem de mais baixo nível, para ajudar na criação de modelos baseados em agentes de maneira mais simples.

Seguindo essa documentação, existem 11 passos direcionam os usuários a se familiarizarem com a ferramenta, aumentando seu nível de complexidade a medida que o usuário avança em seus níveis.

Já o segundo membro da família, o *Repast for High Performance Computing* (*Repast HTC*) tem sua base construída em linguagem c++, foi desenvolvido para simulações em larga escala e utilização em supercomputadores.

É considerado como sendo a nova geração dos sistemas baseados em modelos de agentes, utilizados para computação distribuída. O *Repast HTC* foi projetado para rodar em diferentes sistemas operacionais, buscando assim figurar como opção viável para diversos supercomputadores.

2.1.6.3 GAMA

Ter uma ferramenta mais intuitiva e que pudesse ser utilizado por diferentes áreas de pesquisa, esse foi o conceito inicial por trás do desenvolvimento da ferramenta Gama. Desenvolvida entre os anos de 2007 a 2010 por uma parceria entre pesquisadores vietnamitas e franceses. Atualmente é mantida por um consorcio de parceiros acadêmicos e industriais, sobre a liderança do *UMMISCO* (*'Unité de Modélisation Mathématique et Informatique des Systèmes Complexes*).

Desenvolvido em Java, e com sua linguagem própria de alto nível (GALM), essa ferramenta tem em sua base diversas configurações pré-instaladas para que

qualquer pesquisador inicie simulações mais genéricas sem nenhuma dificuldade.

Equipado com modelos de sistema de informações geográfica (do inglês, GIS), e com base de dados de modelos de orientação o que torna essa ferramenta ideal para a simulações em larga escala e alta complexidade. Com possibilidade de visualizações em 2D, 3D e agentes individuais, o que facilita a observação das simulações e o ajusto em tempo real.

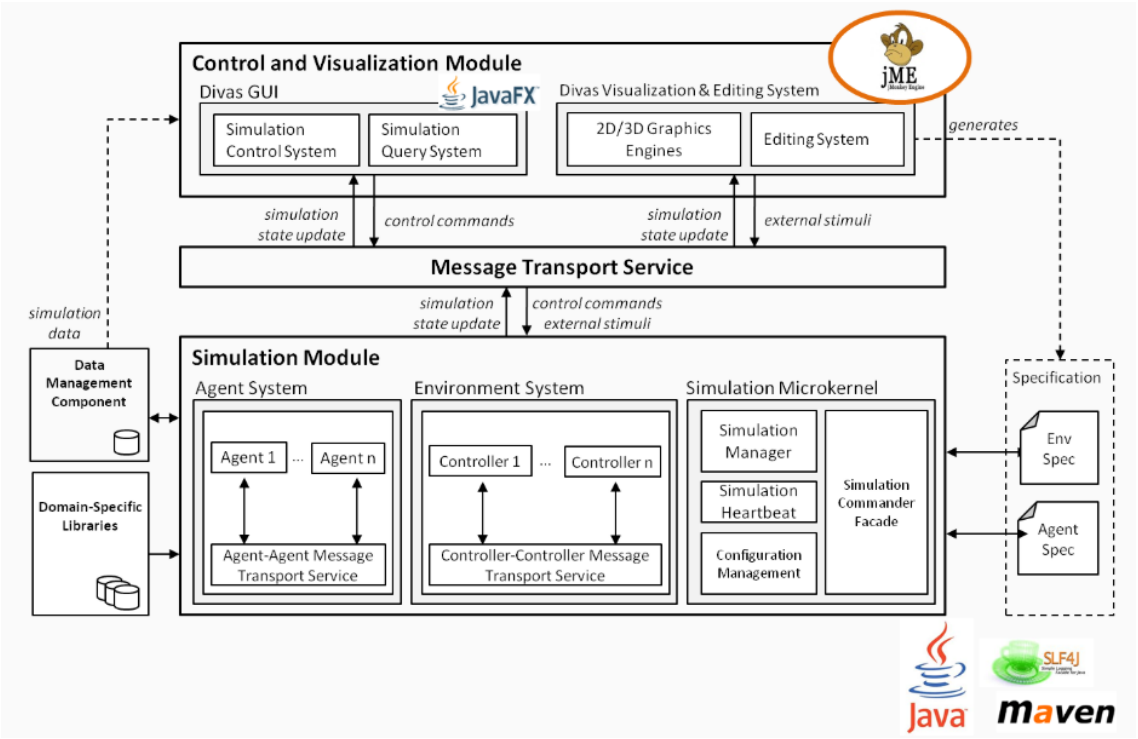
Diversos estudos vêm sendo realizados com essa ferramenta, é importante destacar o estudo realizado nas cidades francesas de Dijon e Grenoble. Essas duas cidades de médio porte foram utilizadas como base para experimentos de simulação de acesso a serviços e pontos estratégicos para a gestão.

2.1.6.4 DIVAS

Essa ferramenta parte do princípio que tanto o agente como o ambiente possuem a mesma importância para a simulação de sistema multiagente. Para tal o DIVAS 4.0 (*Dynamic Information Visualization of Agent systems*), foi desenvolvido para ser uma ferramenta com arquitetura totalmente reutilizável, ou seja, seus componentes de software foram pensados de uma maneira que fiquem independentes e descentralizados.

O DIVAS 4.0 é totalmente implementada em linguagem Java, e seus componentes foram todos pensados nessa linguagem. Uma característica dessa ferramenta é que toda sua construção foi pautada nos princípios da engenharia de software.

Figura 10. Componentes DIVAS 4.0



FONTE: Al-ZANITI at al. (2013)

Na figura 10 temos os principais componentes da aplicação. O módulo de simulação, cria e instancia todas a simulações, já o módulo de transporte fornece a infraestrutura de comunicação para que diferentes elementos da simulação se comuniquem, e por fim o módulo de controle e visualização recebe as informações do módulo de transporte e renderização essas informações em uma visualização 2D ou 3D, além de permitir a interação com a simulação.

Infelizmente a ferramenta em questão não foi disponibilizada ao público em geral. Apenas sua descrição e algumas demonstrações estão disponíveis no site do *The Multi-Agent & Visualization Systems Lab at the University of Texas (*http://www.utdmavs.org/divas/).

2.2 Teoria Dos Jogos

2.2.1 APRESENTAÇÃO

A teoria dos jogos é um ramo da matemática que se preocupa com o comportamento estratégico de agentes racionais, em situação que exigem raciocínio estratégico e interação entre agentes (von NEUMANN & MORGENSTERN, 1972).

Essa teoria é utilizada para estudar diferentes assuntos como eleições, comportamento de mercados, evolução genética, inteligência computacional etc. Sua formalização foi feita pelo matemático húngaro John Von Neumann, que publicou artigos como "*Zur Theorie der Gesellschaftsspiele*" em 1928 e "*A Model of General Economic Equilibrium*" publicado em 1937, dando os primeiros passos. Com a publicação do livro *Theory of Games and Economic Behavior* juntamente com o economista austríaco Oskar Morgenstern, a teoria dos jogos chamou a atenção do público. Desde então vem ganhando força para resolução de problemas onde há interação entre agentes racionais.

2.2.2 COMPONENTES DE UM JOGO DE ESTRATÉGIA

A palavra "jogo" foi alvo de diversos mal-entendidos desnecessários. Muitos acabavam associando a teoria dos jogos com algo frívolo, sem muita importância e nada científico. No entanto, essa má impressão se desfaz ao perceber que o objeto de estudo é simplificar e modelar matematicamente as relações interpessoais que ocorrem rotineiramente (SCHOUERY, 2015).

Em seu trabalho sobre teoria dos jogos e filosofia, Silva (2014) destaca que "para a Teoria dos Jogos, um jogo é definido por um conjunto de regras que estabelece seus cinco elementos constitutivos" que serão enumerados a seguir:

1. Número de jogadores;
2. Ações ou estratégias possíveis;
3. Os resultados de cada jogador;
4. A função que permite a cada parte combinar suas estratégias e
5. A relação de preferência de cada jogador mediante seus resultados.

De maneira mais simples, LUCE e RAIFFA (1957 apud Teixeira 2012, p.152-153) elenca os principais objetos de um jogo de estratégia. Esses elementos são:

- **O jogo:** são representações simplificadas de situações onde pelo menos uma pessoa age no sentido de maximizar a utilidade de suas ações levando em conta as reações de outros (Silva, 2014). Pode ser formalizado utilizando ferramentas matemáticas e possui regras que orientam as interações dos jogadores;

- **Interações:** ações realizadas pelos agentes participantes do jogo, orientadas pelas regras que regem o jogo;

- **Jogadores:** qualquer agente ou grupo de agentes que pode realizar interações dentro do ambiente de jogo;

- **Racionalidade:** comportamento que busca métodos mais adequados parta a obtenção de sucesso na busca por seus objetivos, em outras palavras, comportamento que visa maximizar os ganhos de um agente tido como racional;

- **Comportamento estratégico:** as ações de um determinado jogador afetam as ações de outros jogadores. Ao levar em consideração como suas ações afetam outros, esse jogador apresenta o comportamento que melhor lhe favoreça, ou seja, apresenta um comportamento estratégico acerca dos outros jogadores. Existem dois tipos de estratégias de comportamento que são: as estratégias puras, onde os jogadores sempre agem da mesma maneira; e, as estratégias mistas, onde os jogadores a partir das interações mesclam estratégias puras (TEIXEIRA, 2012).

2.2.3 TIPOS DE JOGOS

Para que uma situação possa ser considerada um jogo, ela teria que apresentar uma situação de conflito e interdependência entre as decisões dos participantes. De uma maneira mais concreta, podemos identificar dois tipos de

jogos: (1) os **jogos não-cooperativos**, são as interações onde os jogadores não podem ou não querem formar alianças com outros jogadores; e, (2) **os jogos cooperativos**, onde a dinâmica do jogo permitem a formação de alianças entre dois ou mais jogadores;

Os **jogos não cooperativos**, também são chamados de **jogos de soma-zero**. Nesse modelo de jogo, os jogadores buscam tomar decisões para benefício próprio. Nesse tipo de jogo pode haver um número finito ou infinito de estratégias ou alternativas de interação. Associado as estratégias a um valor de pagamento (*payoff*) ao seu oponente (FIGUEIREDO, 1994).

Do contrário, quando esta situação não ocorre é dito que o jogo não é mais estritamente competitivo e a soma das funções de utilidade dos jogadores não resulta mais em zero. Nesse caso, o jogo passa a ser denominado de **jogo de soma-não-zero** (LUCE e RAIFFA, 1957 apud TEIXEIRA, 2012). Um exemplo desse tipo de jogo é o dilema do prisioneiro que será tratado com mais detalhes na próxima seção.

Em 1949, Jonh F. Nash Jr. propôs em sua tese de doutorado um teorema com foco em jogos não-cooperativos de soma-não-zero, esse teorema ficou mundialmente conhecido como equilíbrio de Nash.

Em resumo, todo o jogo com uma tabela de pagamento (*payoff*) possui um ponto de equilíbrio onde todos os jogadores ganham valores iguais e que nenhum poderia melhorar seus ganhos com decisões unilaterais (NASH JR, 1950).

Os jogos chamados cooperativos constituem uma classe de jogos que se diferencia dos jogos não-cooperativos, pela existência em sua estrutura interna de condições que favorecem a formação de aliança entre jogadores (FIGUEIREDO, 1994).

Sobre os jogos cooperativos Teixeira (2012) afirma que os jogos cooperativos são aqueles onde os jogadores se comunicam antes do início da disputa e, assim, podem definir algum tipo de acordo em relação às estratégias que serão utilizadas no jogo.

2.2.4 O PARADIGMA DO DILEMA DO PRISIONEIRO

2.2.4.1 APRESENTAÇÃO

Como exemplo da aplicação da teoria dos jogos, podemos citar o problema apresentado pelos cientistas Melvin Dresher e Merril Flood da RAND Corporation conhecido como Dilema do Prisioneiro. Ele define uma situação de conflito de interesses, onde dois indivíduos são presos e colocados em celas diferentes e sem comunicação entre eles (POUNDSTONE, 1992).

2.2.4.2 DILEMA PARA DUAS PESSOAS

2.2.4.2.1 SIMPLES

Albert W. Tucker adaptou o problema original, adicionando a questão do tempo da sentença de prisão. O Dilema do Prisioneiro clássico funciona da seguinte forma: Dois suspeitos, A e B, são presos pela polícia. A polícia tem provas insuficientes para os condenar, mas separadamente é ofertado a ambos o mesmo acordo: se um dos prisioneiros, confessando, testemunhar contra o outro e esse outro permanecer em silêncio, o que confessou sai livre enquanto o cúmplice silencioso cumpre 10 anos de sentença. Se ambos ficarem em silêncio, a polícia só pode condená-los a 6 meses de cadeia cada um. Se ambos traírem o comparsa, cada um leva 5 anos de cadeia. Cada prisioneiro faz a sua decisão sem saber que decisão o outro vai tomar, e nenhum tem certeza da decisão do outro.

A tabela 1, apresenta os pagamentos para o dilema do prisioneiro, onde é apresentado todas as combinações possíveis para a interação dos jogadores.

Tabela 1. Tabela de pagamento do dilema do prisioneiro

		J_B	
		Cooperar (C)	**Trair (D)**
J_A	**Cooperar (C)**	(R, R)	(S, T)
	Trair (D)	(T, S)	(P, P)

FONTE: Adaptado de Borges (1996)

Observando a tabela 1, pode-se verificar que cada um dos jogadores tem dois tipos de comportamentos: Cooperar(C) e Trair (D). Além disso, tem-se quatro variáveis que mudam de acordo com a recompensas. Elas são:

(1) R é a recompensa (*Reward*) para cada jogador caso ambos venham a cooperar;

(2) P é a punição (*Punishment*) para cada jogador caso ambos venham a trair;

(3) T é a tentação (*Temptation*) de cada um dos jogadores, caso traíssem sozinhos;

(4) S é o “pagamento do otário” (*Sucker*) que coopera sozinho.

Para que seja configurado um dilema do prisioneiro, essas quatro variáveis devem obedecer às seguintes condições:

$$\text{(I)}\quad T > R > P > S$$

$$\text{(II)}\quad R > \frac{T+S}{2}$$

$$\text{(III)}\quad \frac{T+S}{2} > S$$

2.2.4.2.2 INTERADO

O problema original do dilema do prisioneiro consistia em apenas uma iteração enquanto os prisioneiros estavam separados. Mas qual seria a consequência para os jogadores se eles soubessem o que seu oponente fez e se ele pudesse escolher novamente?

Pautado nessa pergunta, o Dilema do Prisioneiro Iterado propõe “n” iterações entre os jogadores, sendo que as ações anteriores podem ser consultadas e utilizadas no processo de tomada de decisão.

Esse novo modo de pensar o Dilema do prisioneiro tradicional, deu início a diversos trabalhos em que o ponto de estudo é o comportamento dos jogadores depois dessas interações. Alguns exemplos são: AXELROD (1984), POUNDSTONE (1992), BORGES (1996), KENDALL, YAO, CHONG (2007), KNIGHT (2017).

2.2.4.3 DILEMA PARA N-PESSOAS

Por mais que o Dilema do Prisioneiro venha sendo ponto focal de pesquisa nas últimas décadas, essa modelagem apresenta dificuldades em algumas situações reais, principalmente em problemas sociais e que envolvam mais de dois jogadores (TEIXIERA, 2012).

Buscando englobar problemas de mais de dois jogadores, foi desenvolvido o conceito do Dilema do Prisioneiro para N-Pessoas. Essa proposta é definida por três propriedades elementares (KENDALL, YAO, CHONG (2007), TEIXEIRA (2012)):

1. Cada jogador pode escolher entre os comportamentos de cooperar e trair;
2. O comportamento de traição é dominante para cada um dos jogadores, ou seja, é melhor trair do que cooperar não importando quantos outros jogadores tenham feito a opção de cooperar;
3. As estratégias dominantes se cruzam em um ponto de equilíbrio deficitário, ou seja, se todos escolherem trair o resultado será pior do que se todos tivessem escolhido a cooperação.

Jonh Nash (1949) em seu trabalho *Equilibrium points in n-person games,* apresenta o modele matemático para o ponto de equilíbrio onde todos os jogadores permanecem com a mesma expectativa.

As funções de pagamento podem ser obtidas generalizando a matriz de pagamento do Dilema do Prisioneiro original, levam em consideração o jogador que coopera (J_C) e os jogadores que traem (J_D).

$$J_C = \frac{R(N_C - 1)}{(N_J - 1)} \qquad (2.2)$$

$$J_D = \frac{(T \times N_C + (N_J - N_C - 1)}{(N_J - 1)} \qquad (2.3)$$

Onde:

- R é a recompensa;
- T é a tentação;
- N_c é a quantidade de jogadores que cooperaram na rodada atual;
- N_J é a quantidade total de jogadores que estão participando da competição.

De maneira geral, a tabela 2 apresenta a tabela de pagamento dessa abordagem do Dilema do Prisioneiro.

Tabela 2. Tabela de pagamento Dilema do Prisioneiro N-pessoas

		Quantidade de outros jogadores, que escolheram o comportamento Cooperar (C)						
		0	**1**	**2**	**3**	**...**	**N-2**	**N-1**
J₁	**C**	$N_C = 1$ S	$N_C = 2$ $\frac{3}{(N_J - 1)}$	$N_C = 3$ $\frac{6}{(N_J - 1)}$	$N_C = 4$ $\frac{9}{(N_J - 1)}$	...	$\frac{R(N_J - 2)}{(N_J - 1)}$	R
	D	$N_C = 0$ P	$N_C = 1$ $\frac{(N_J + 3)}{(N_J - 1)}$	$N_C = 2$ $\frac{(N_J + 7)}{(N_J - 1)}$	$N_C = 3$ $\frac{(N_J + 11)}{(N_J - 1)}$	...	$\frac{(T(N_J - 2) + 1)}{(N_J - 1)}$	T

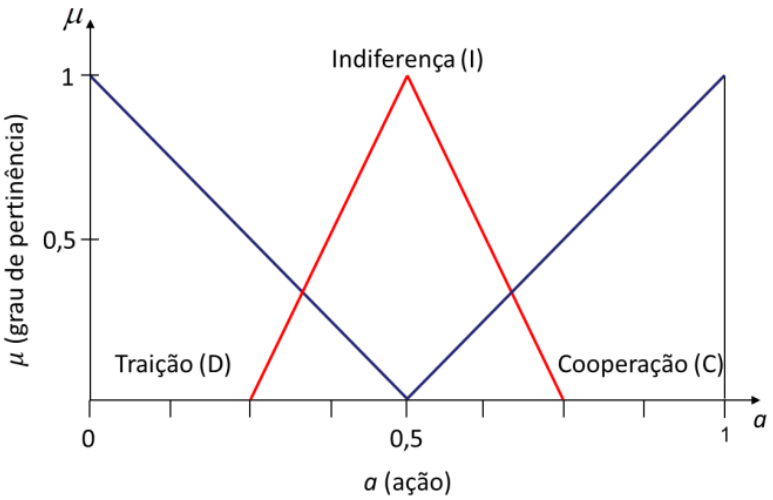
FONTE: Adaptado de Manhart (2007)

Pode-se observar que para o caso de N_J=2, temos a tabela de pagamento do dilema do prisioneiro original.

2.2.4.4 DILEMA DO PRISIONEIRO NEBULOSO

O Dilema do Prisioneiro Nebuloso se diferencia do modelo clássico que aceita apenas o conjunto binário de cooperação (C) e Traição (D). Nessa abordagem do jogo, as interações dos jogadores são mapeadas em um intervalo [0,1], ou seja, o comportamento do agente varia de (0) traição total até (1) cooperação total.

Figura 11. Gráfico do possível comportamento dos indivíduos no FIPD



FONTE: Borges (1996)

Baseado no grau de pertinência $\mu_C(a)\ ou\ \mu_D(a)$ é definido o nível de cooperação/traição de cada jogados. Esse conjunto nebuloso foi subdividido em dois subconjuntos mutuamente exclusivos de Traição(D) e Cooperação (C). Além de um terceiro subconjunto denominado indiferente(I), que consiste em uma ação de "tanto faz" cooperar ou trair.

As equações (2.4) e (2.5), definem o grau de pertinência da ação do jogador nos subconjuntos de Traição(D) e Cooperação (C).

$$\mu_D(a) = \begin{cases} -2a + 1, & a \leq 0{,}5 \\ 0, & a \geq 0{,}5 \end{cases} \tag{2.4}$$

$$\mu_C(a) = \begin{cases} -2a + 1, & a \geq 0{,}5 \\ 0, & caso\ contrário \end{cases} \tag{2.5}$$

A partir dos graus de pertinência obtidos nessas duas equações é possível calcular o valor da ação de cada jogados (a_j), como demostrado na equação (2.6).

Equação 1

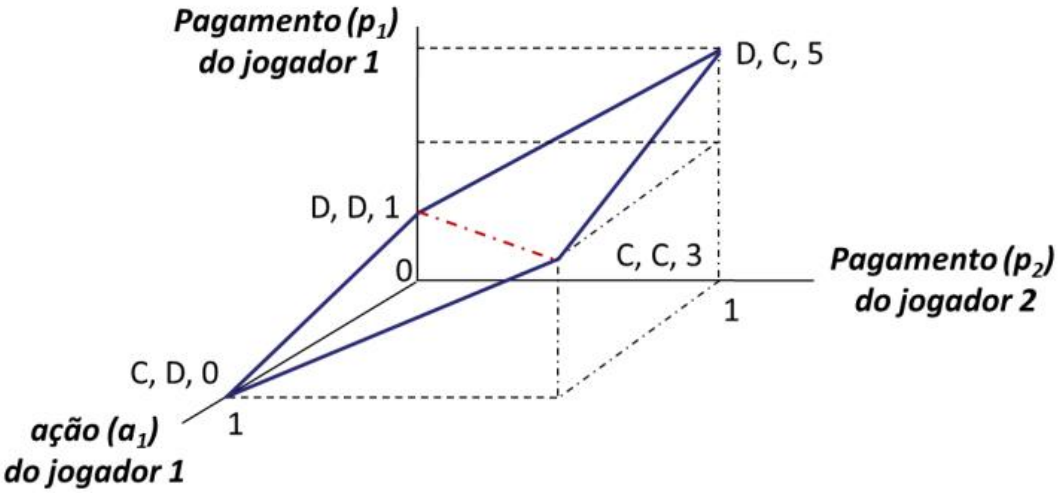
$$a_j = \frac{\left((1 - \mu_D(a)) \times \mu_D(a) + (1 - \mu_c(a)) \times \mu_c(a)\right)}{2(\mu_D(a) + \mu_c(a))} \tag{2.6}$$

A tabela de pagamento clássica é substituída por uma função de pagamento, com duas variáveis independentes ($a_1 \; e \; a_2$), e que geram o pagamento do jogador ($J_1 \; e \; J_2$). Os valores de pagamento são baseados nas ações externas de cooperação (C) e traição, além $\mu_C(a) = 1 \; e \; \mu_D(a) = 0$, assim como T=5, R=3, P=1 e S=0. Logo os fatotes de ganho podem ser expressos pelas equações (2.7) e (2.8) e a função linear representada na figura 12.

$$p_{j_1} = \begin{cases} 1 - 2a_1 + 4a_2, & a_1 < a_2 \\ 1 - a_1 + 3a_2, & a_1 \geq a_2 \end{cases} \qquad (2.7)$$

$$p_{j_2} = \begin{cases} 1 - 2a_2 + 4a_1, & a_2 < a_1 \\ 1 - a_2 + 3a_1, & a_2 \geq a_1 \end{cases} \qquad (2.8)$$

Figura 12. Gráfico da função de pagamento do FIPD



FONTE: Borges (1996)

2.2.5 TORNEIOS COMPUTACIONAIS

Na década de 70, o cientista político Robert Axelrod da University of Michigan, propôs um torneio baseado nos fundamentos do famoso dilema do prisioneiro. Esse torneio consistia em colocar diferentes estratégias de comportamento para disputar (dois a dois) em partidas de 200 movimentos (AXELROD, 1984).

Aquino (2012) ressalta que os torneios computacionais promovidos por Axelrod, tem um papel fundamental na evolução do conhecimento gerado sobre a inteligência computacional baseada em interação com outros agentes.

2.2.5.1 PRIMEIRO TORNEIO – AXELROD

O primeiro torneio foi realizado com 14 estratégias apresentadas pelos participantes e mais a estratégia aleatória (RANDOM). Para a surpresa dos participantes a estratégia vencedora, foi considerada a mais simples apresentada (AXELROD, 1984).

Segundo Axelrod (1984), a estratégia apresentada pelo psicólogo matemático Anatol Rapoport era considera a mais simples entre todas as estratégias. Denominada de TIT FOR TAT (olho por olho), foi a grande vencedora da disputa, utilizando um padrão de começar cooperando e depois repetir o último movimento de seu oponente.

Na tabela 3 temos todas as estratégias presentes no primeiro torneio com os respectivos nomes de seus autores, quantidade de linhas de código em Fortran e a pontuação obtida no referido torneio.

Tabela 3. Estratégias do primeiro torneio computacional de Axelrod

Posição Final	Nome do Autor	Linhas de código	Pontuação
01	Anatol Rapoport	04	504.5
02	Nicholas Tideman & Paula Chieruzzi	41	500.0
03	Rudy Nydegger	23	485.5
04	Bernard Grofman	08	481.9
05	Martin Shubik	16	480.7
06	Willian Stein & Ammon Rapoport	50	477.8
07	James W Friedman	13	473.4
08	Morton Davis	06	471.8
09	James Graaskamp	63	400.7
10	Leslie Downing	33	390.6
11	Scott Feld	06	327.6

12	Johann Joss	05	304.4
13	Gordon Tullock	18	300.5
14	Name withheld	77	282.2
15	RANDOM	05	276.3

FONTE: Autor (2019)

2.2.5.2 SEGUNDO TORNEIO – AXELROD

Devido ao grande sucesso decorrente do primeiro torneio computacional do dilema do prisioneiro e os horizontes de pesquisa abertos pelo mesmo. Na década de 80 Axelrod realizou o segundo torneio computacional.

Nesse torneio foram apresentadas 62 estratégias e mais a opção aleatória (RANDOM), incluindo algumas estratégias já utilizadas no primeiro torneio. Axelrod (1984) destaca que muitas estratégias presentes nesse segundo torneio foram preparadas especificamente para vencer a estratégia TIT FOR TAT. Fato que não se consolidou, novamente a estratégia de Rapoport se manteve em primeiro lugar.

Os torneios computacionais realizados por Axelrod se tornaram um marco na pesquisa de comportamento. Halás (2011) expõe que os torneios computacionais de Axelrod marcaram uma nova era no que diz respeito a utilização da teoria dos jogos para modelagem de ambientes de disputa. Teoria essa que por muito tempo foi deixada de lado por grande parte da comunidade cientifica.

2.2.5.3 TORNEIO NEBULOSO – BORGES

Na década de 90, utilizando a abordagem nebulosa do Dilema Prisioneiro, Borges (1996) realizou uma série de torneios para o estudo do tema. Esses torneios foram o ponto de partida para diversas outras pesquisas como BORGES (1996; 1997), SONG e KANDEL (1999), RAMAN (2002), MATHEW e KAIMAL (2003), e ARFI (2006).

O torneio implementado por Borges foi escrito em C++ e contou com 512 jogadores utilizando estratégias nebulosas, que foram divididos em 32 grupos com 16 participantes cada. Em cada grupo foram introduzidos 3 jogadores com estratégias não difusas, TFT tradicional, TFT generalizado e o Pavlov. Depois da formação dos grupos foram realizadas três fases.

Na primeira fase, os 32 grupos com seus 19 integrantes (16 nebulosos e 3 não), tiveram 30 000 iterações e assim como nos torneios originais de Axelrod, foram classificados de acordo com seu resultado médio. Na segunda fase, foram escolhidos os oito melhores de cada grupo e formaram onze grupos novos.

Na terceira fase, foram escolhidos os jogadores com melhor performance para formar novos 5 grupos. Dessa fase saíram os 16 melhores e foram agrupados juntos com as três estratégias não nebulosas para uma disputa final. Em todas as fases foram realizadas 30 000 iterações, isso quer dizer que os jogadores se encontraram com o mesmo adversário em torno de 166 vezes.

À medida que as fases foram avançando, houve um domínio massivo do jogador com a estratégia TFT, na primeira fase a estratégia TFT não venceu em nenhum dos 32 grupos, já na segunda fase dos 11 grupos, 8 apresentavam o TFT como primeiro colocado. Na terceira fase todos os 5 grupos obtiveram o TFT como sendo o primeiro colocado e na iteração final, novamente o TFT foi o vencedor absoluto.

2.2.5.4 TORNEIO DE 20 ANOS DO DILEMA DO PRISIONEIRO

Em 2001, durante o Congresso de Computação Evolucionaria, surgiu a ideia de ser realizado em 2004 a comemoração do aniversário de 20 anos dos torneios computacionais realizados por Axelrod em 1984.

Esse torneio ocorreu em duas etapas, a primeira no ano de 2004 contou com 3 competições e o segundo em 2005 contou com apenas uma competição. Na primeira fase, foi permitido que um participante pudesse apresentar mais de uma estratégia, isso gerou um grande discursão, devido a um grupo de participantes ter preparado uma quantidade de estratégias que se comportavam como mestre e outras como escravos e que tinha a “habilidade” de se reconhecerem.

1. Na primeira competição de 2004, foi utilizado a configuração padrão do Dilema do Prisioneiro Iterado, a mesma utilizada por Axelrod;
2. Na segunda competição, foi introduzido um ruído, que consistia em uma probabilidade de 0,1 em que a decisão podia ser mal interpretada;
3. A competição três consistia de um jogo para n-pessoas, ou seja, tinha uma tabela de pagamento diferente do jogo original.

Devido aos problemas apresentados no primeiro torneio, em 2005, na Conferência de Inteligência Computacional e Games foi proposto uma nova competição, dessa vez cada participante poderia apresentar apenas uma única estratégia.

Baseado nas quatro competições, foi lançado o livro “The Itarated Prisoners’ Dilemma: 20 years on”, contendo os principais discursões do torneio. O proponente de cada estratégia vencedora pode apresentar um artigo para os capítulos do livro. Além do livro, foi criado um website(http://www.prisoners-dilemma.com/) para orientar e agrupar as ações pertinentes a essa competição.

Por fim, é importante ressaltar que foi desenvolvido uma ferramenta para simulação de torneios computacionais. Mais precisamente para pesquisas futuras sobre o torneio de 20 anos.

2.2.5.5 TORNEIO PYTHON – KNIGHT

Em 2016, Vincent Knight iniciou um projeto que se tornou a base para o presente trabalho. Por enfrentar uma grande dificuldade para ressimular os torneios anteriores, foi desenvolvido uma biblioteca em Python que reuni as principais estratégias dos torneios anteriores.

O torneio de Knight se baseia em uma versão modificada do Dilema do Prisioneiro tradicional, uma vez que no torneio original as estratégias buscavam maximizar seus ganhos, no torneio de Knight as estratégias buscam minimizar seus ganhos, a tabela 4 mostra a tabela de pagamento desse torneio.

Tabela 4. Tabela de pagamento do torneio Knight

		Jogador 2	
		Cooperar	Trair
Jogador 1	Cooperar	(2,2)	(5,0)
	Trair	(0,5)	(4,4)

FONTE: Adaptado de Knight (2017)

Esse torneio foi composto por 129 estratégias, que foram desenvolvidas pelos mais diferentes públicos, desde alunos do ensino médio até equipes de centros de pesquisa. A grande diferença nesse torneio está na composição das estratégias. Além das estratégias tradicionais, foram implementadas estratégias baseadas em

modelo de máquinas de estados finitos que utilizam parâmetros de algoritmo evolucionário ou enxame de partículas.

Todo o torneio foi documentado, é importante ressaltar essa preocupação pois é um dos objetivos do projeto que esse torneio e os demais que se sucederem depois possam ser totalmente reprodutíveis. Para tal, toda a aplicação é disponibilizada e com seu código FONTE aberto, além de uma extensa documentação organizada em uma plataforma web (https://axelrod.readthedocs.io/en/stable/index.html).

2.2.6 TRABALHOS CORRELATOS

Buscar uma ferramenta que consiga simular os torneios do dilema do prisioneiro iterado não é uma tarefa fácil, no decorrer dos anos diversas ferramentas foram desenvolvidas, assim como bibliotecas contendo as estratégias de alguns torneios.

É necessário ressaltar que o único trabalho que encontramos com desenvolvimento contínuo foi o proposto por Knight (2017), que apesar de ser uma ferramenta formidável, apresenta algumas limitações que serão explanadas mais a frente.

Com o intuito de comparação abordaremos três ferramentas desenvolvidas com a finalidade de ressimular os torneios computacionais que aconteceram durante o decorrer da história. Essas ferramentas, em ordem cronológica, são : (1) WINIPD desenvolvida em 2003 por Chris Cook, como uma ferramenta para que qualquer pessoa pudesse montar seu próprio torneio; (2) IPD TOURNAMENT , desenvolvida em 2004/2005 para a celebração dos 20 anos do Dilema do Prisioneiro Iterado; e (3) A biblioteca desenvolvida por Knight(2017), com o intuito de agrupar as estratégias dos primeiros torneios e simular torneios com estratégias baseadas em aprendizagem de máquina.

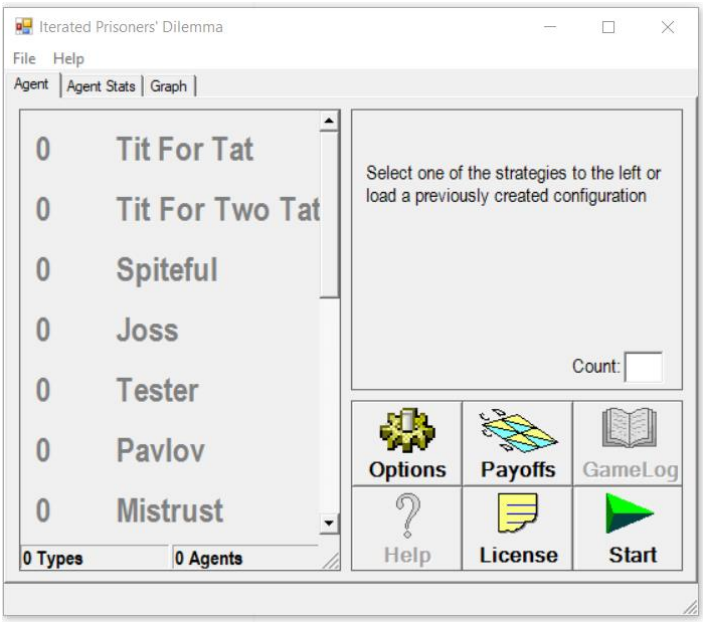
Os critérios utilizados para a comparação das fermentas serão os apresentados por Cook(2011) e Kendall, Yao e Chong (2007), que estão enumerados a seguir: (1) Linguagem de programação; (2) Interface de usuário

(GUI); (3) Quantidade de estratégias implementadas; (4) Quantidade de iterações suportadas; (5) Tabela gráfica; (6) Registro e tratamento dos torneios.

2.2.6.1 WinIPD

Uma das ferramentas mais interessantes do gênero, possui uma interface simples (Figura 13), e não tão intuitiva. Foi desenvolvida em Java e possui uma licença de código aberto.

Figura 13. Interface WinIPD

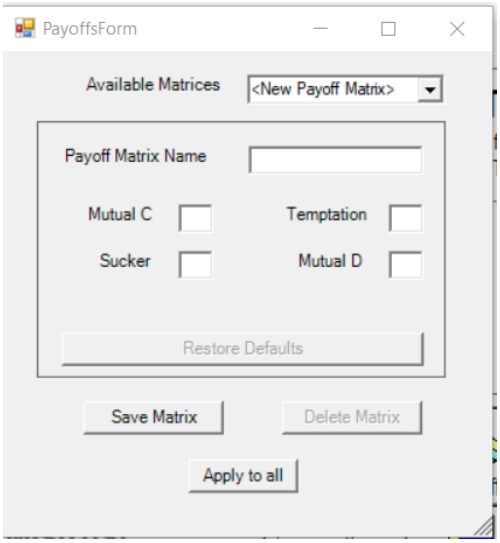


FONTE: Cook (2011)

Apresenta como funcionalidade interessante a possibilidade de alterar a matriz de pagamento (Figura 14), mas não apresenta restrições ou alertas sobre a forma correta de configurar a matriz, levando a inconsistências durante a execução.

Esta pré-configurada para realizar 120 iterações entre os participantes, podendo realizar mais dependendo da quantidade de estratégias selecionadas. São definidas 16 estratégia de comportamentos que podem ser usadas para a criação de torneios, podendo escolher a quantidade de participantes até o limite das estratégias.

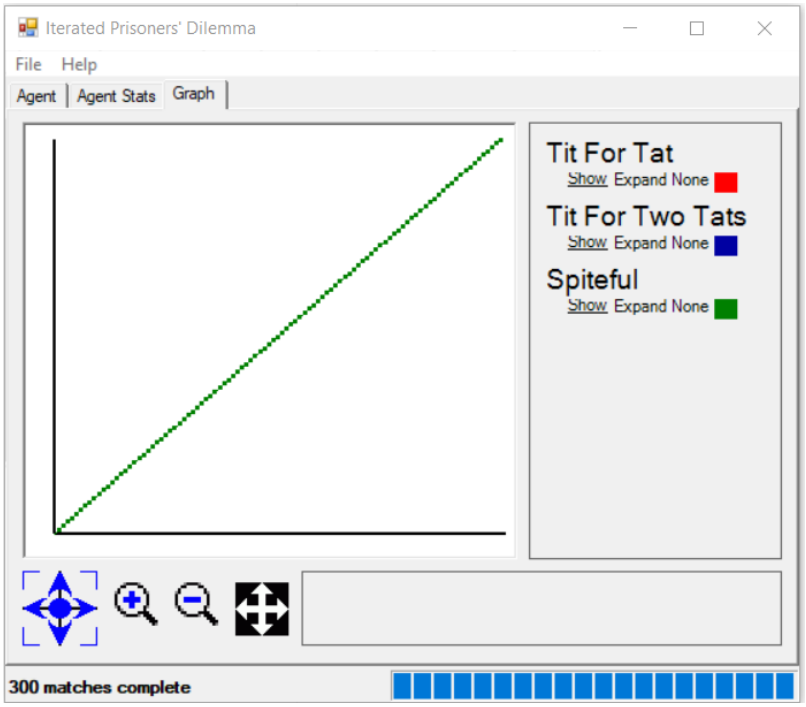
Figura 14. Matriz de pagamento



FONTE: Cook (2011)

Possui uma interface de apresentação dos torneios realizados, e possui a função de exportar para arquivo CVS. Além de possuir uma função de estatística dos agentes e a visualização dos gráficos de evolução das iterações entre os agentes.

Figura 15. Janela de gráficos do WinIPD



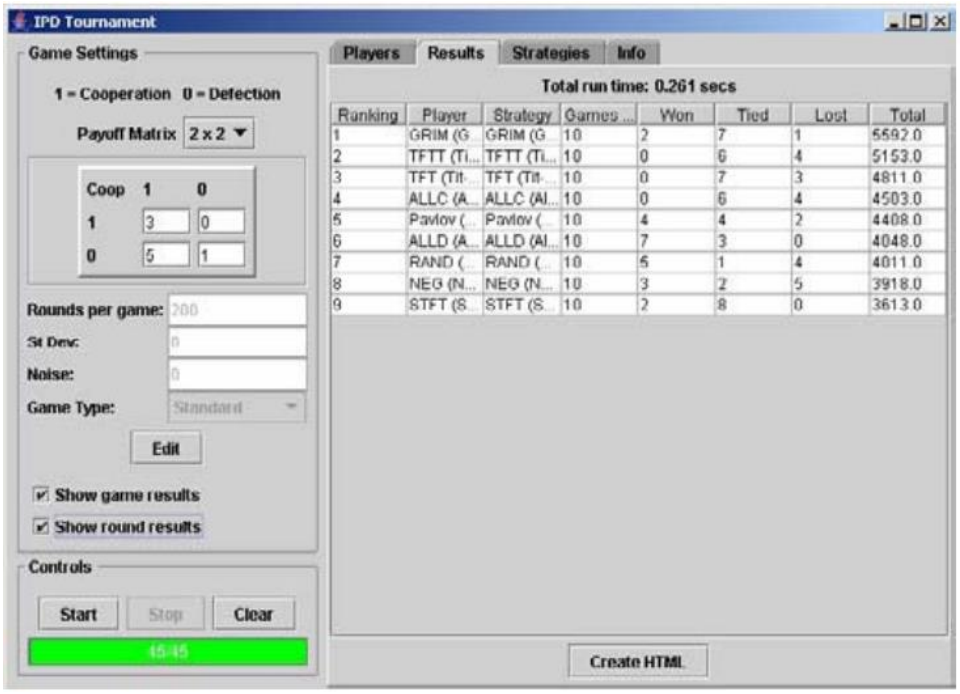
FONTE: Cook (2011)

2.2.6.2 IPD TOURNAMENT

Ferramenta desenvolvia em Java, seu código-fonte não foi disponibilizado para estudo. Foi criada especificamente para o torneio dos 20 anos e trabalhos futuros. Na presente data o link do repositório onde a ferramenta estava disponível para download encontra-se desativado.

Possui uma interface intuitiva, com opção de alteração na tabela de pagamento, opção de inserir ruído nos torneios, além de apresentar de forma clara os resultados dos torneios realizados.

Figura 16. Interface de usuário IPD Tournament



FONTE: Kendall, Yao e Chong (2007)

Foram implementadas as principais estratégia presentes na data de desenvolvimento, sua biblioteca conta com 16 estratégia iniciais para utilização dos usuários, sua interface não permite a inclusão de estratégias desenvolvidas pelos usuários.

Dentro da aplicação existe a opção de apresentar o gráfico do comportamento das estratégias, mas não permite a exportação em formato cvs, o que possibilitaria edição em ferramentas de planilhas.

2.2.6.3 Biblioteca Knight

Essa biblioteca não possui interface gráfica, mas é de extrema importância para o presente trabalho. Desenvolvida em Python 3.6, possui uma licença de código aberto, além de possuir uma extensa documentação muito bem organizada, disponível em https://axelrod.readthedocs.io/en/stable/index.html.

Possui 242 estratégias em seu repositório e apresenta a funcionalidade de ressimular o primeiro e o segundo torneio de Axelrod. Não possui limite de iterações e nem limite de estratégias participantes no torneio. Implementado já vem uma função que chama todas as estratégias presentes na biblioteca e realiza um torneio com todas.

Não apresenta a funcionalidade de exportação em CVS, e nem a função de gráficos. Seus relatórios são bastante limitados, uma vez que sua apresentação é feita em um terminal de comandos.

Outro ponto negativo da biblioteca é a falta de algumas estratégias do segundo torneios de Axelrod, que ainda não foram implementadas em Python. Além de em determinadas execuções apresentar algumas inconsistências quanto a tabela de pagamento. Tabela essa que já vem pré-definida e pode ser alterada.

Essa biblioteca é a primeira do tipo, tanto me relação a quantidade de estratégias, quanto a preocupação de reunir em um só lugar as condições primordiais para a realização de estudos sobre o Dilema do Prisioneiro iterado.

2.2.6.4 Comparativo entre as ferramentas

Na tabela 5 é apresentado de forma resumida as principais características das ferramentas acima ditadas.

Tabela 5 - Comparação entre as ferramentas de simulação dos torneios computacionais

FERRAMENTA	WinIPD	IPD TOURNAMENT	Biblioteca Knight
Linguagem de programação	JAVA	JAVA	PHYTON
Interface de	SIM	SIM	NÃO

usuário (GUI);			
Quantidade de estratégias implementadas	16	16	200
Quantidade de iterações suportadas	ILIMITADO	120	ILIMITADO
Tabela gráfica	SIM	SIM	NÃO
Registro e tratamento dos torneios	SIM	NÃO	SIM

FONTE: Autor (2019)

3 PROTÓTIPO DE SOFTWARE PARA O DILEMA DO PRISIONEIRO ITERADO

3.1 Síntese

O trabalho de Russell e Norvig vem ao longo dos anos se consolidando como uma das principais referências, do que tange ao tema Inteligência artificial. Neste trabalho, é apresentado um olhar diferenciado do capítulo dois do livro, que aborda o conceito estruturas de agentes.

Costume-se pensar na estrutura de agentes como sendo uma trajetória linear que vai de sua estrutura mais básica, um agente reativo simples até sua estrutura mais elevada, um agente baseado em aprendizado. Esse pensamento não está incorreto uma vez que cada agente possui um complemento evolutivo em relação ao seu antecessor.

Figura 17. Modelo linear das estruturas de agentes

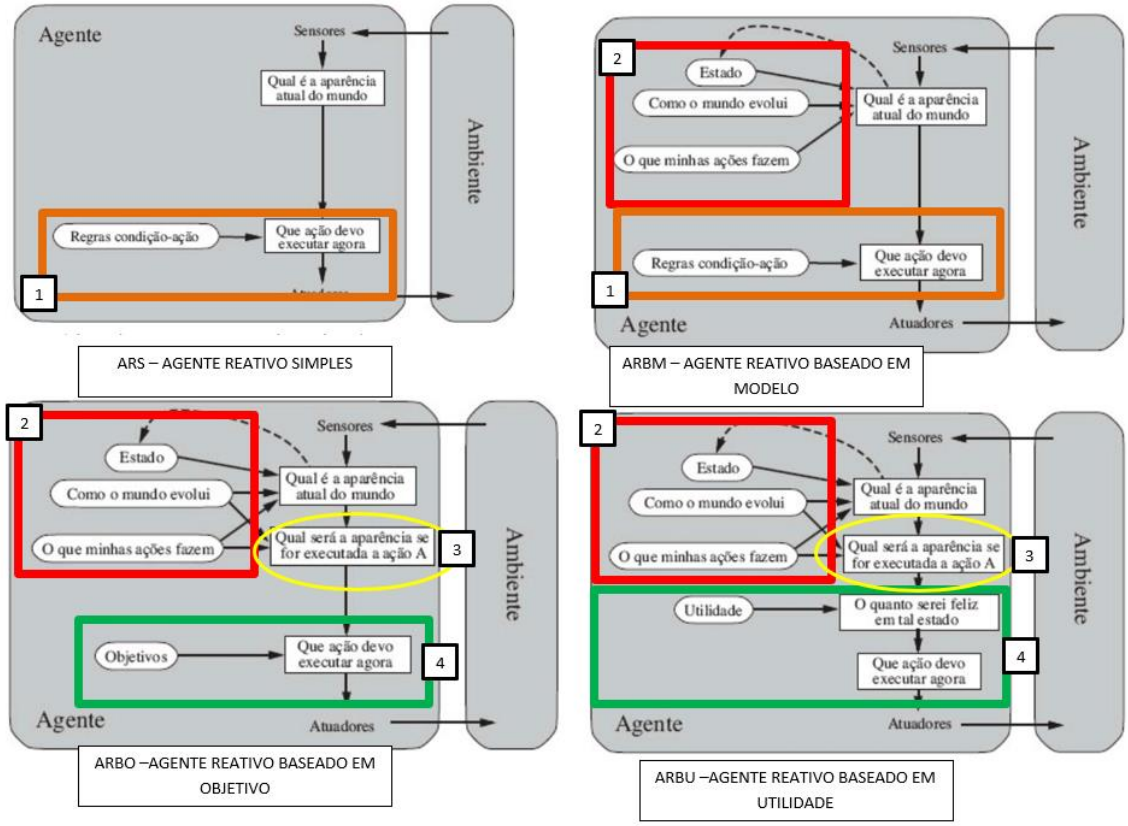
FONTE: Autor (2019)

Na figura 17, é feita a representação linear das estruturas de agentes, cada estágio tem sua particularidade. Como cada estrutura é a evolução da anterior, podemos ressaltar dois pontos:

1. Tomando o agente reativo simples como base, cada estrutura seguinte surgiu devido as necessidades impostas pelo ambiente, ou seja, ambientes mais complexos necessitam de agentes mais complexos;
2. Cada estrutura carrega “comportamentos” pertencentes a suas antecessoras.

Os dois pontos citados são complementares, devido ao processo de evolução dos agentes. Ao ser analisado cada ponto específico do comportamento dos agentes, podemos perceber como as estruturas são adicionadas e/ou repetidas como mostrado na figura 18.

Figura 18. Comparação entre as estruturas de agentes



FONTE: Adaptada de Russell e Norvig (2013)

As estruturas de agentes foram agrupadas em quatro funções semelhantes de acordo com sua utilidade, essa separação foi realizada buscando facilitar a visualização do processo de evolução de cada arquitetura, mostrando elos de semelhança das estruturas anteriores.

A função um (1) apresenta o princípio de toda a ação efetiva do agente, um simples condicional onde toda a informação que é recebida e “tratada” e devolvida para ser executada. Essa função pode ser percebida de maneira mais efetiva nos agentes reativos simples e nos baseados em modelos, mas não deixa de existir nos agentes mais complexos, apenas é englobado pela função 4.

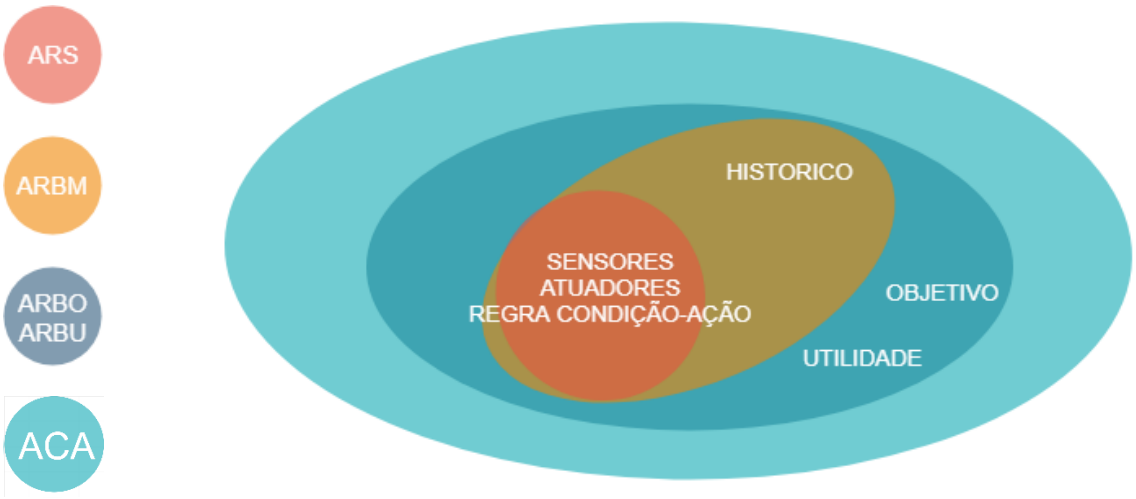
A função dois é adicionada a partir dos agentes baseados em modelo, e de forma simplificada ela pode ser considerada como um histórico das iterações do agente, seja com o ambiente ou com qualquer outro agente. Nos modelos baseados em objetivo e utilidade essa função é diretamente atrelada a função três (3), que cria a percepção de como as ações do agente influenciam o ambiente.

A função quatro(4) é voltada para a obtenção do objetivo, com a pequena variação de no agente de utilidade a busca ser focada em cumprir o objetivo da melhor forma, mas isso não tira o fato que a busca é por um objetivo especifico e isso foi tomado como verdade para simplificar a classificação das estratégias de comportamentos.

Por fim, a junção de todas essas funções são a base para a construção do agente baseado em aprendizagem. O **Crítico,** faz uso da função dois(2) e três(3), tanto o **elemento de aprendizagem**, quanto **elemento de desempenho** utilizam a função quatro(4) em seus processos.

Em decorrência dessa divisão foi gerado um diagrama que possibilita a compreensão do grau de pertencimento de cada função a diferentes estruturas.

Figura 19. Representação em conjunto das estruturas de agentes



FONTE: Autor (2019)

Essa separação fez-se necessária para facilitar a classificação das estratégias de comportamento presentes no tópico seguinte.

3.2 Classificação das Estratégias Baseado no Programa do Agente (CEBPA)

Somando todas as estratégias presentes nos 5 torneios que compõem este trabalho, foram levantadas 754 estratégias. Knight (2017) destaca que devido ao grande potencial dos estudos realizados no campo da teoria dos jogos, mas especificamente nos estudos de comportamento utilizando o dilema do prisioneiro e

suas diferentes vertentes, a quantidade de estratégias de comportamento é incalculável.

Não existe uma forma de classificação específica voltada para essas estratégias de comportamento, até para geração de novos torneios as estratégias são escolhidas arbitrariamente pelo pesquisador que está desenvolvendo a pesquisa, geralmente para torneios iterados as estratégias utilizadas para validação, são as estratégias dos primeiros torneios de Axelrod.

Neste trabalho é feita a proposta de classificação das estratégias de comportamento utilizando como base os tipos de agentes propostos por Russell e Norvig (2013), partindo dos seguintes pressupostos:

1. Um agente é composto por uma estrutura e seu programa;
2. Uma estratégia de comportamento não é diferente de um programa, ou seja, ela é a descrição de como um agente deve agir dada suas percepções;
3. A estrutura de um agente utilizado para estudos de comportamento é fixa, composta pela percepção de entrada da ação de cooperar ou trair dos jogadores no ambiente, mais seu comportamento e sua ação de saída.

Como apresentado no capítulo 2, um agente pode ser classificado de acordo com um dos 5 tipos a seguir: ARS, ARBM, ARBO, ARBU e ACA. Cada estratégia de comportamento também pode ser classificada de acordo com essas estruturas tomando como base seu programa.

Para a classificação foram estipuladas as seguintes condições:

1. Memoria: Toda a estratégia que precisa salvar mais de uma iteração é caracterizada como estratégia com memoria;
2. Uma única estratégia pode pertencer a diferentes grupos de classificação, baseado no número de iterações que serão realizadas;
3. Objetivo: Uma única métrica a ser atingida pela estratégia;
4. Utilidade: Diversas métricas e funções na construção da estratégia;
5. Aprendizado: Utiliza-se das iterações anteriores e mais um conjunto de funções para compreender que estratégia o oponente está utilizando.

Observando-se as condições acima citadas, é possível associar cada agente a um conjunto de características que são usadas para a classificação das estratégias. A seguir é apresentado os agentes e suas características principais para classificação.

ARS → Não possui memória e suas ações pré-determinadas;

ARBM → Tem memória e capacidade apenas de identificar os comportamentos anteriores dos oponentes e seguir uma regra pré-determinada;

ARBO → Tem memória e possui um ou mais estados desejados, que são formas de alcançar o objetivo;

ARBU → Tem memória e possui um ou mais estados desejados, além de possui um mecanismo (função de avaliação) para medir o grau de felicidade. Usa funções e probabilidades para alcançar os estados;

ACA → Tem memória; consegue decidir seu comportamento "observando" e analisando o comportamento dos outros jogados; tem a capacidade de descobrir que comportamentos o outro jogador está utilizando.

Para exemplificação, a tabela 6 apresenta a classificação do primeiro torneio de Axelrod, em nosso protótipo todas as estratégias presentes nos cinco torneios já estão classificadas.

Tabela 6. Classificação das estrategias de comportamento do primeiro torneio de Axelrod

Posição Final	Nome do Autor	Descrição	Classificação
01	Anatol Rapoport	Na primeira rodada o jogador cooperar e a partir da segunda rodada ele repete o movimento anterior de seu oponente.	ARBM
02	Nicholas Tideman & Paula Chieruzzi	Essa estratégia tem uma inicialização semelhante ao TFT [4], a mudança ocorre na resposta às traições. Na primeira traição ocorre a resposta de retaliação simples do TFT, caso a cooperação seja retomada e o oponente volte a trair o jogador retribui com uma traição e incrementa mais uma traição na sequência (n), caso o oponente continue traindo é feita a retribuição normal do TFT, voltando a ser retomada a cooperação e o oponente voltando a trair o jogador retribui com o número de traições anterior incrementando mais uma (n+1) persistindo a traição a resposta e a mesma do TFT. Além do comportamento citado acima existe chama de "fresh start" ou novo	ARBU

		recomeço. Ela ocorre se o oponente tiver uma sequência de duas colaborações consecutivas e estiver com 10 pontos a menos que jogador [6]: E não comece a trair de novo; E já tenha se passado 20 rodadas desde a última "fresh start"; E existe mais de 10 rodadas para terminar o torneio; E o número de traições tem desvio padrão diferente de 3.0 em uma amostra aleatória 50 – 50.	
03	Rudy Nydegger	Começa com a estratégia TFT para as três primeiras rodadas com exceção se na primeira rodada ele for o único a cooperar e na segunda rodada for o único a trair. Nesse caso na terceira rodada ele irá trair novamente. A partir desse momento as próximas decisões serão baseadas nas três rodadas anteriores seguindo as condições abaixo: Cada traição do oponente será atribuído um valor de 2; Cada traição do jogador será atribuída um valor de 1; O valor resultante da soma entre a pontuação atribuída as traições (n_1) na primeira das três rodadas que estão sendo avaliadas terá o peso 16, o valor (n_2) da segunda rodada um peso 2 e o valor (n_3) da terceira um peso 1, resultando na seguinte fórmula: $A = 16n_1 + 4n_2 + n_3$ Sendo que o jogador apenas trairá se o resultado de A= {1, 6, 7, 17, 22, 23, 26, 29, 30, 31, 33, 38, 39, 45, 49, 54, 55, 58, 61} caso contrário, o jogador irá cooperar.	ARBU
04	Bernard Grofman	O jogador inicia cooperando. Caso seu oponente tenha feito o oposto, na próxima roda a uma chance de $\frac{2}{7}$ para cooperação. Caso as ações sejam iguais ele sempre coopera.	ARBM
05	Martin Shubik	Essa estratégia tem uma inicialização semelhante ao TFT, a mudança ocorre na resposta as traições. Na primeira traição ocorre a resposta de retaliação simples do TFT, caso a cooperação seja retomada e o oponente volte a trair o jogador retribui com uma traição e incrementa mais uma traição na sequência (n), caso o oponente continue traindo é feita a retribuição normal do TFT, voltando a ser retomada a cooperação e o oponente voltando a trair o jogador retribui com o número de traições anterior incrementando mais uma (n+1) persistindo a traição a resposta e a mesma do TFT.	ARBM / ARBU
06	Willian Stein & Ammon Rapoport	Inicia cooperando durante as 4 primeiras rodadas e depois assume o comportamento TFT, avaliando a cada 15 rodadas pelo teste de qui-quadrado se o oponente está jogando	ARBM / ARBU

		randomicamente. Caso esteja ele só trai.	
07	James W Friedman	Essa estratégia colabora até que seu oponente traía, daí pra frente começa a só trair até o final do jogo	ARBM
08	Morton Davis	Coopera nas 10 primeiras rodadas, caso o oponente traía, o jogador começa a trair até o final do jogo	ARBM
09	James Graaskamp	Essa estratégia segue as seguintes regras: Utiliza a estratégia TFT nas 50 primeiras rodadas; Na rodada número 51 ele traí; Nas próximas 5 rodadas ele utiliza TFT; Depois é feita a verificação se o oponente está jogando de forma aleatória, caso positivo, o jogador começa a trair até o final do jogo; Caso seja constatado na verificação que o oponente está utilizando a estratégia TFT ou alguma estratégia semelhante o jogador assume TFT até o final do jogo; Caso contrário ele coopera e traí de forma aleatória a cada 5 ou 15 rodadas.	ARBM / ARBU/ ACA
10	Leslie Downing	Atualiza duas probabilidades ao longo das jogadas p (cooperações do oponente | cooperações do jogador) e p (cooperações do oponente | traições do jogador) e seleciona a ação que a longo prazo tem maior expectativa de retorno. Inicialmente as duas probabilidades iniciam em 0,5	ARBU
11	Scott Feld	Inicia utilizando o TFT, é diminui a probabilidade de cooperação após a cooperação do adversário na rodada anterior de maneira que na iteração 200 esse valor é de 0,5.	ARBO
12	Johann Joss	Inicia cooperando e joga TFT, com a diferença de que quando o oponente coopera ele tem uma probabilidade de 90% de cooperar.	ARBM
13	Gordon Tullock	Coopera nas 11 primeiras rodadas, depois aleatoriamente coopera 10% menos do que o oponente cooperou nas últimas 10 rodadas.	ARS / ARBU
14	Name withheld	Essa estratégia coopera baseada em uma probabilidade que inicialmente e igual a 0.3 e é atualizada a cada 10 rodadas baseada se o oponente e aleatório, muito colaborativo ou pouco colaborativo.	ARBU
15	RANDOM	Coopera ou traí com uma igual probabilidade, ou seja, 0.5.	ARS

FONTE: Autor

Pelo período desse torneio, década de 80, é relativamente evidente o prevalecimento de estratégias que são classificadas como ARBM E ARBU, no decorrer dos torneios é perceptível a mudança de paradigma, uma vez que com o passar dos anos as estratégias tendem aos agentes com aprendizado.

No torneio realizado por Knight (2017) é predominante a presença das estratégias classificadas como agentes com aprendizado, a própria constituição do torneio busca estudar esse tipo de estratégias.

Na próxima sessão é apresentado um protótipo de ferramenta intuitiva que busca implementar de maneira simples e objetiva essa visão de classificação, além de ser uma ferramenta para o estudo de torneios computacionais.

3.3 Descrição da Ferramenta

O protótipo desenvolvido neste trabalho permite a ressimulação, de maneira simples e intuitiva, dos Torneios Computacionais de Axelrod (1984); Kendall, Yao e Chong (2004); Borges (1996) e Knight (2012). Permite também a aplicação do conceito desenvolvido e discutido neste trabalho, além de admitir a configuração de novos torneios podendo neste escolher quais estratégias, turnos e a tabela de pagamento a serem utilizados. Nesse cenário, a tabela de pagamento é considerada o ambiente no qual os agentes irão interagir.

Todo o desenvolvimento foi realizado utilizando a linguagem Python na sua versão 3.6, pois foi observado o crescente uso da linguagem para a codificação em áreas como *Big Data, Machine Learning* e Inteligência Artificial. Somando ao fato de se tratar de uma linguagem de multiplataforma – tendo funcionamento e performance igualmente em diversos sistemas operacionais – e sua grande comunidade de desenvolvedores. Para a construção da interface gráfica foi utilizado o pacote *PyQt* e o software *Qt Designer.*

O projeto foi construído em quatro (4) etapas. Na primeira etapa foi desenvolvida a classe FIPD responsável por implementar o código da abordagem do Dilema do Prisioneiro de Borges, onde foram implementadas 512 estratégias que originalmente na linguagem *C++*.

Para o seu funcionamento, primeiramente são carregados os jogadores, após esse passo serão gerados os trinta e dois (32) grupos da primeira fase para realizar as iterações, em seguida é gerado uma lista de todos os pares de encontros que serão efetuados naquela fase, após esse passo será percorrido a lista inteirando cada par entre si. A partir deste ponto, o programa segue a mesma sequência lógica

descrita na seção 2.2.5.3. O usuário poderá modificar a quantidades de iterações, como será mostrada na figura 24.

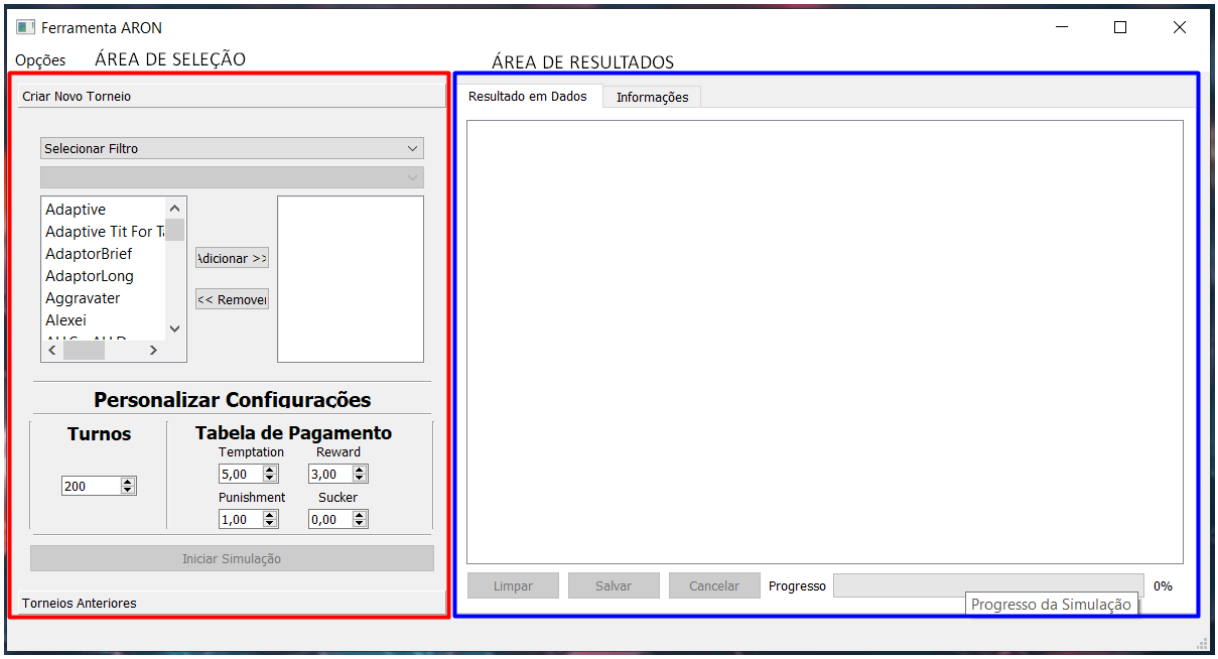
Na segunda etapa, foram organizadas as estratégias codificadas por Knight de acordo com os torneios as quais foram criadas – Axelrod e Kendall Yao e Chong – por meio da implementação de um pacote gerenciador de torneiros. Uma vez que não era possível, até então, ter uma relação das estratégias por torneios anteriores, o que tínhamos era uma robusta biblioteca com mais de 200 estratégias que só permitia a chamada individual de cada estratégia ou a listagem de todas as estratégias implementadas. Essa característica implicava diretamente nas ressimulações, para obter os dados de um torneiro anterior era necessário chamar cada estratégia utilizada no torneio e configurar as iterações dois a dois das estratégias.

Na terceira etapa, foram organizadas, na aplicação, as estratégias dos cinco Torneio Computacionais referenciados neste trabalho de acordo com a classificação criada na seção 3.2.

Na quarta etapa foi construído a interface gráfica da ferramenta. O intuito foi trazer uma aplicação simples e ao mesmo tempo fácil de ser utilizada, tendo apenas uma janela, e com poucos cliques o usuário possa configurar seu próprio torneio, simular torneios anteriores, modificando ou não seu ambiente, além de poder salvar os dados gerados em arquivos com extensão *xls*.

Quando a ferramenta é iniciada temos uma tela dividida em duas áreas, como mostrado na figura 20, a área a esquerda da tela, destacada em azul, é composta por duas (2) abas nas quais são responsáveis por obter as configurações de agentes e ambiente que o usuário configure. A área da direita, destacada em vermelha, é o campo que exibirá os dados gerados das simulações em formatado de tabela ranqueando as estratégias que obtiveram os melhores ganhos.

Figura 20. Tela do Protótipo com Áreas Destacadas

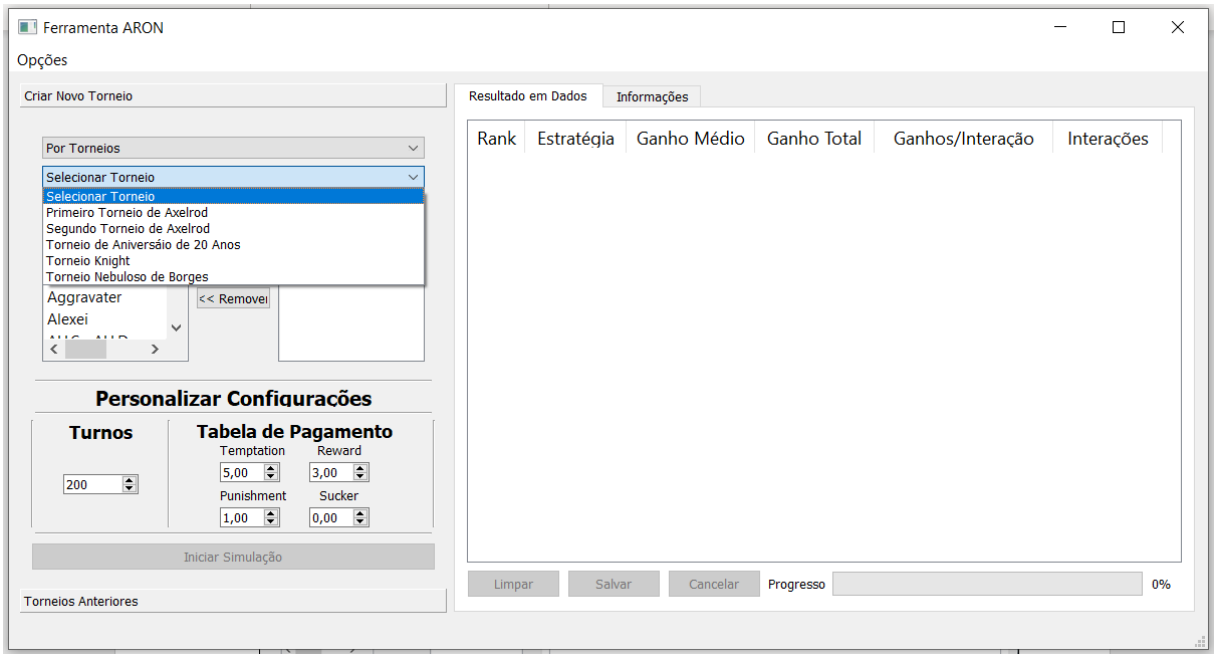


FONTE: Autor (2019)

Explanando de melhor forma a área esquerda da ferramenta, têm-se duas abas, na aba superior da tela é o espaço para a configuração de um novo torneio, no qual é possível utilizar qualquer estratégia implementada na aplicação, assim como configurar a sua própria tabela de pagamento. No primeiro momento é mostrado todas as estratégias implementadas na ferramenta, porém têm-se dois filtros através de caixas de seleção para a chamada das estratégias, os filtros auxiliam a escolha, organizando a exibição das estratégias por torneios computacionais ou pela CEBPA (seção 3.2).

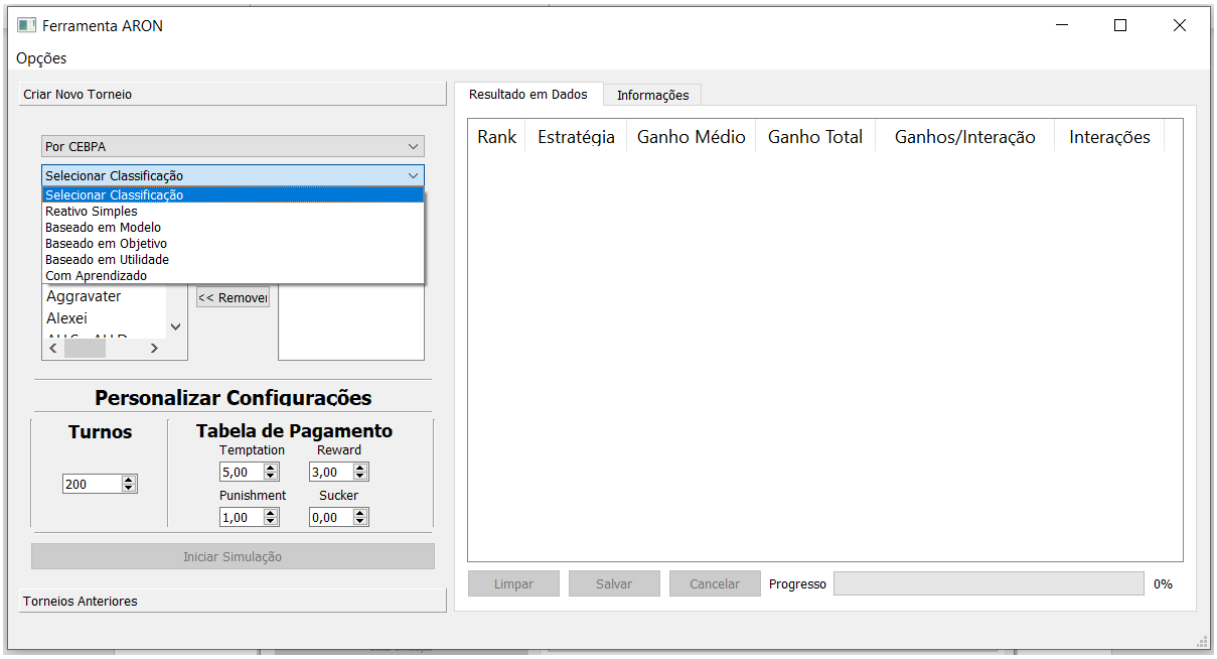
As figuras 21, 22 e 23 mostram o funcionamento das caixas de seleção. Abaixo da seleção, têm-se a personalização das configurações, com a tabela de pagamento e a quantidade de turnos a serem realizados. Após o usuário selecionar os agentes e configurar o ambiente é possível iniciar o novo torneio clicando no botão *Iniciar Simulação.*

Figura 21. Tela do Protótipo com Destaque ao Primeiro Filtro de Seleção



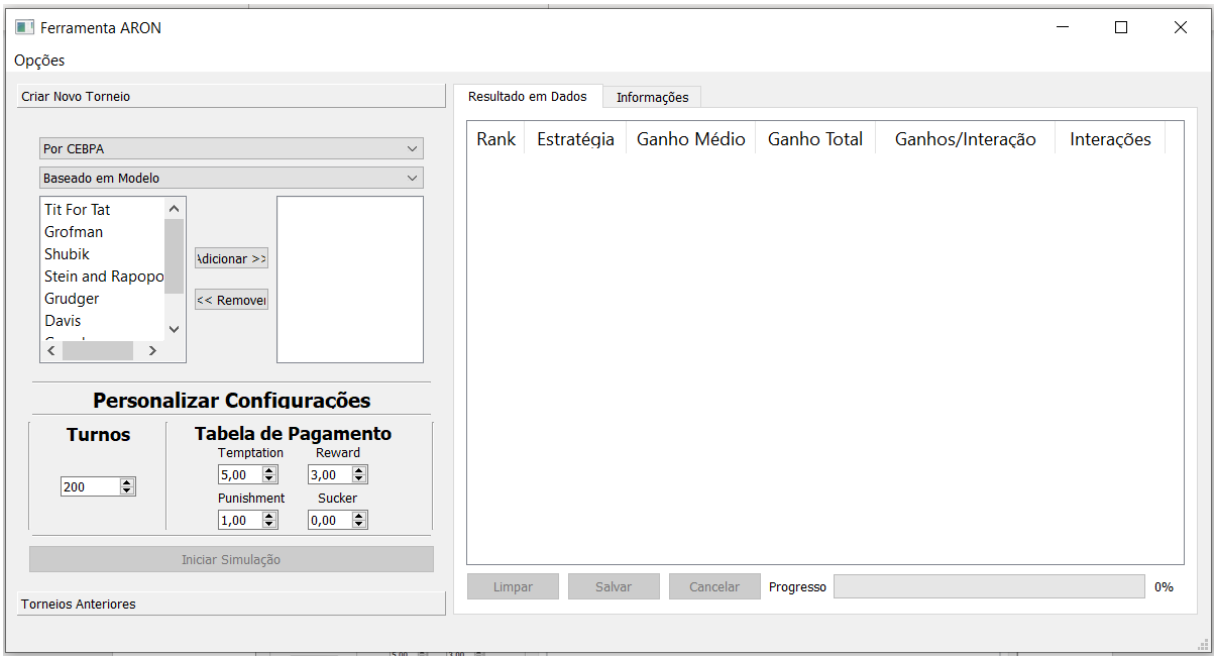
FONTE: Autor (2019)

Figura 22. Tela do Protótipo com Destaque ao Segundo Filtro de Seleção Por Torneio



FONTE: Autor (2019)

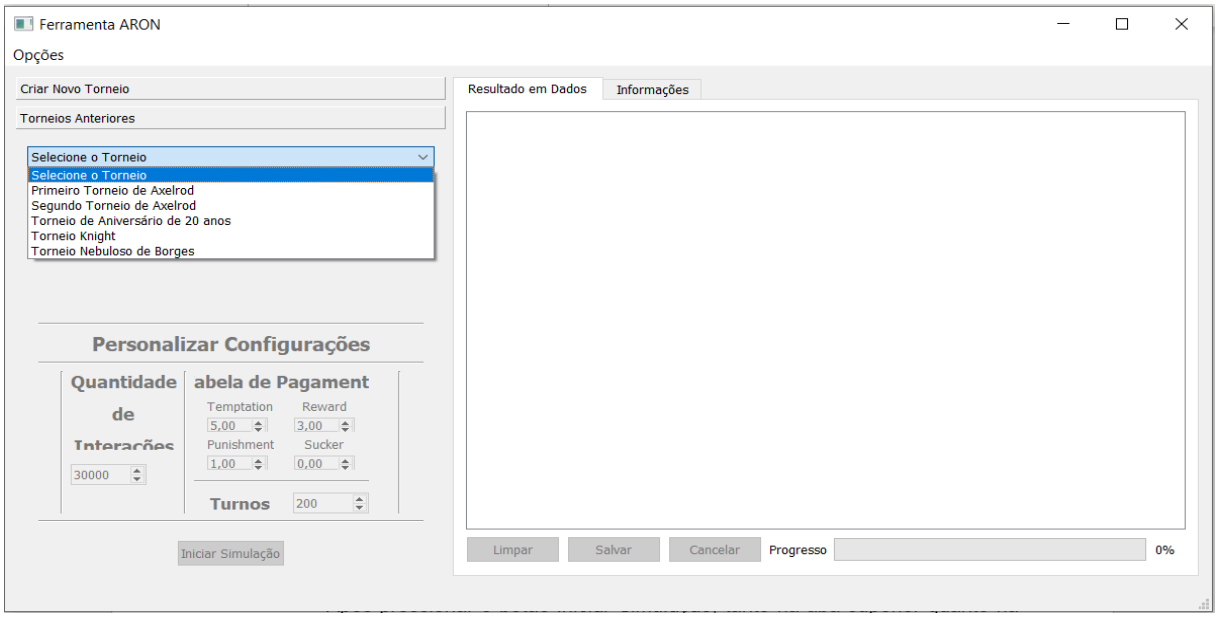
Figura 23. Tela do Protótipo com Destaque ao Primeiro Filtro de Seleção Por Classificação



FONTE: Autor (2019)

Na aba inferior da tela, figura 24, possibilita a ressimulação dos torneios anteriores. Nessa aba temos um caixa de seleção com os nomes de cada torneio, abaixo temos a tabela de pagamento, uma entrada para a quantidade de iterações, caso seja a ressimulação do torneio nebuloso e uma entrada para o número de turnos, caso seja a ressimulação dos demais torneios, a ferramenta inicia com a tabela de pagamento padrão utilizada, caso o usuário queira obter os dados de acordo com os torneios já realizados é necessário apenas selecionar o torneio e clicar no botão *Iniciar Simulação.*

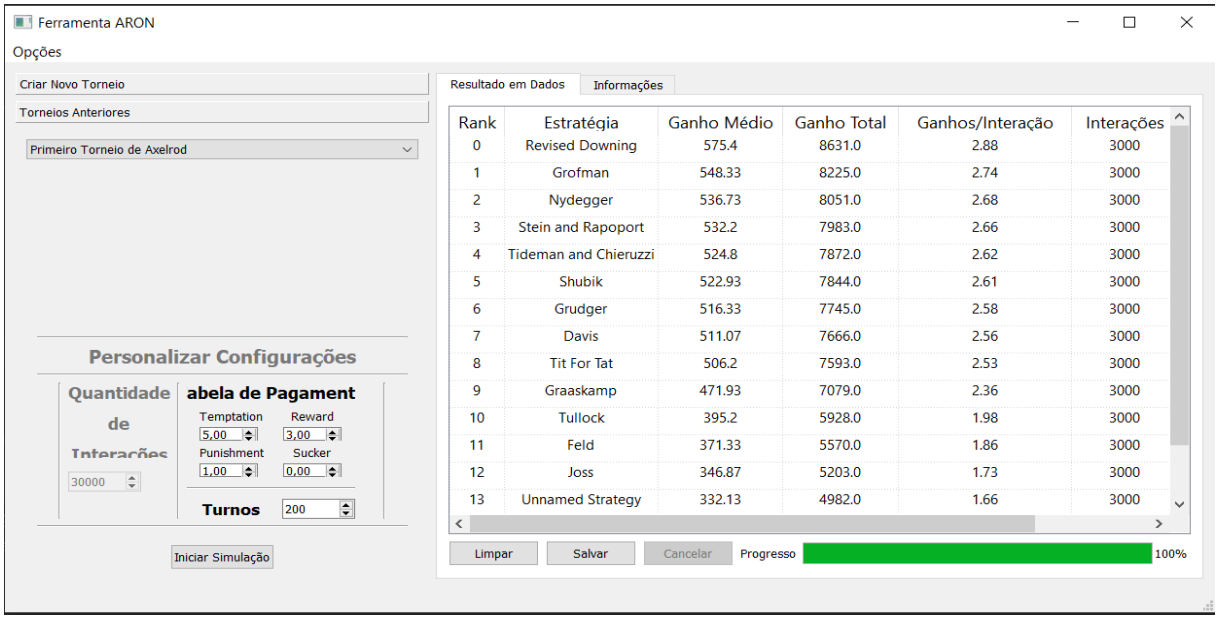
Figura 24. Tela do Protótipo - Aba Para Simular Torneios Anteriores



FONTE: Autor (2019)

Após pressionar o botão *Iniciar Simulação,* tanto na aba superior quanto na inferior, será iniciado o torneio programado, a barra de progresso exibirá a porcentagem na qual o torneio está em sua execução, caso o usuário queira abortar a simulação basta pressionar o botão *Cancelar.* Após o encerramento da simulação os dados irão aparecer na área direita, *Resultado em Dados,* conforme a figura 25. É possível salvar os dados clicando no botão *Salvar* e limpar a tela no botão *Limpar.*

Figura 25. Tela de Resultados



FONTE: Autor (2019)

4 AVALIAÇÃO DO PROTÓTIPO

O projeto do protótipo seguiu uma linha de desenvolvimento iniciando com a síntese das estruturas propostas por Russell e Norvig. O passo seguinte foi o levantamento e a classificação das estratégias de comportamentos dos 5 principais torneios e pôr fim a implementação de toda a pesquisa em uma ferramenta simples e objetiva.

Nessa seção, será apresentado no item 4.1 a comparação da ferramenta desenvolvida, em relação aos trabalhos correlatos mencionados na seção 2.2.6. No item 4.2 serão apresentados os resultados das simulações do primeiro torneio de Axelrod, o troneio nebuloso de Borges e o torneio mais recente realizado por Knight.

4.1 Comparação do Protótipo com Trabalhos Correlatos

Como relatado anteriormente, os critérios utilizados para a comparação das fermentas serão os apresentados por Cook(2011) e Kendall, Yao e Chong (2007), que estão enumerados a seguir: (1) Linguagem de programação; (2) Interface de usuário (GUI); (3) Quantidade de estratégias implementadas; (4) Quantidade de iterações suportadas; (5) Tabela gráfica; (6) Registro e tratamento dos torneios.

Tabela 7. Comparação entre as ferramentas e protótipo

FERRAMENTA	WinIPD	IPD TOURNAMENT	Biblioteca Knight	Protótipo ARON
Linguagem de programação	JAVA	JAVA	PHYTON	PHYTON
Interface de usuário (GUI);	SIM	SIM	NÃO	SIM
Quantidade de estratégias implementadas	16	16	200	754
Quantidade de iterações suportadas	ILIMITADO	120	ILIMITADO	ILIMITADO
Tabela/ Gráficos	SIM	SIM	NÃO	SIM
Registro e tratamento dos torneios	SIM	NÃO	SIM	SIM

FONTE: Autor (2019)

A tabela 7, apresenta a comparação entre as principais ferramentas utilizadas como base para o trabalho e a protótipo desenvolvido. É importante ressaltar que o protótipo atente todos os quesitos elencados por Cook (2011) e Kendall, Yao e Chong (2007).

A escolha da linguagem busca suprir as necessidades futuras, uma vez que essa ferramenta é apenas o pontapé inicial de um projeto mais robusto e que ainda levará alguns anos para estar finalizado.

O fato de implementar tanto os torneios convencionais, como a base para os torneios Nebulosos é um dos diferenciais dessa ferramenta em relação as outras ferramentas apresentadas. Assim como o registro minucioso de qualquer torneio criado, formando uma base de dados para futuros estudos.

Durante cada passo do desenvolvimento, foi levado em consideração uma diretiva básica, essa diretiva busca que a aplicação seja o mais simples possível sem perder suas funcionalidades. Essa preocupação tem como base o público alvo que essa ferramenta procura atender, estudantes e pesquisados que queiram ressimular os torneios computacionais, e/ou criar seus próprios torneios, sem a necessidade de saber programar.

Em resumo, o protótipo é leve, simples, desenvolvido em uma linguagem de programação atual, com uma base gigantes de estratégias para serem utilizadas. Reuni em um só lugar as principais características das ferramentas utilizadas até o momento, além de agrupar os principais torneios realizados nos últimos 30 anos.

4.2 Resultado das Ressimulações dos Torneios Computacionais

4.2.1 Ressimulação do Primeiro Torneio de Axelrod

Durante as ressimulação do primeiro torneio, foi utilizada a biblioteca desenvolvida por Knight, ela apresentou problemas em algumas estratégias o que ocasionou um resultado bastante diferente dos divulgados por Axelrod no seu livro.

Outros fatores influenciam a mudanças dos valores, principalmente devido a muitas estratégias apresentarem comportamentos probabilísticos e/ou comportamentos randômicos dependendo das interações com seus oponentes.

Tabela 8. Ressimulação do 1º Torneio de Axelrod

Rank	Estratégia	Ganho Total	Ganho Médio	Ganho/Iteração	Iterações
0	Revised Downing	8675	578,3333333	2,891666667	3000
1	Grofman	8128	541,8666667	2,709333333	3000
2	Nydegger	8092	539,4666667	2,697333333	3000
3	Tideman and Chieruzzi	7871	524,7333333	2,623666667	3000
4	Shubik	7809	520,6	2,603	3000
5	Stein and Rapoport	7804	520,2666667	2,601333333	3000
6	Grudger	7757	517,1333333	2,585666667	3000
7	Davis	7724	514,9333333	2,574666667	3000
8	Tit For Tat	7610	507,3333333	2,536666667	3000
9	Graaskamp	6985	465,6666667	2,328333333	3000
10	Tullock	6075	405	2,025	3000
11	Feld	5599	373,2666667	1,866333333	3000
12	Joss	5055	337	1,685	3000
13	Unnamed Strategy	4964	330,9333333	1,654666667	3000
14	Random	4918	327,8666667	1,639333333	3000

FONTE: Autor (2019)

A estratégia TFT, chegou a um valor de ganho médio bem próximo do apresentado no primeiro torneio que foi de 504 pontos (tabela 8), as estratégias que ficaram acima do TFT apresentaram ganhos discrepantes em relação a simulação original.

Estudos adicionais estão sendo realizados para identificar quais estratégias estão apresentando problemas, uma vez que que no torneio original a estratégias TFT foi a campeã absoluta. Para identificar esses tipos o problema a ferramenta passará por uma modificação e contará com uma visualização rodada por rodada tanto na representação de Traição e Cooperação, como também representação numérica dos ganhos.

4.2.2 Ressimulação do Torneio Nebuloso de Borges

Essa ressimulação não apresentou nenhuma surpresa, as fases descritas por Borges (1996) ocorreram todas de uma forma similar. No torneio original, na primeira fase o TFT não aparece em nenhum dos primeiros lugares, em nossa ressimulação essa estratégia aparece em dois grupos.

Por se tratar de uma versão nebulosa, que implica na utilização de graus de pertinências e uma situação aceitável, nas demais fases ocorre como descrito na seção 2.2.5.3, culminando com a vitória do TFT.

Tabela 9. Ressimulação do Torneio Nebuloso de Borges

Rank	Estratégia	Ganho Total	Ganho/Adversário	Ganho/Iteração	Iterações
Fase 1					
1° do Grupo 01	354	8071,056	?	2,638462	3059
1° do Grupo 02	536	7871,735	?	2,450727	3212
1° do Grupo 03	653	8092,318	?	2,579636	3137
1° do Grupo 04	364	7422,491	?	2,405994	3085
1° do Grupo 05	PAVLOV	7534,741	?	2,441588	3086
1° do Grupo 06	443	8457,408	?	2,692585	3141
1° do Grupo 07	655	8981,856	?	2,84416	3158
1° do Grupo 08	343	7816,558	?	2,48935	3140
1° do Grupo 09	644	9471,844	?	2,95349	3207
1° do Grupo 10	346	9438,31	?	2,926608	3225
1° do Grupo 11	633	7719,434	?	2,540123	3039
1° do Grupo 12	636	8464,047	?	2,607531	3246
1° do Grupo 13	PAVLOV	7861,762	?	2,477706	3173
1° do Grupo 14	445	7682,082	?	2,399901	3201
1° do Grupo 15	356	7221,249	?	2,274409	3175
1° do Grupo 16	456	8190,156	?	2,567447	3190
1° do Grupo 17	PAVLOV	7347,344	?	2,296045	3200
1° do Grupo 18	454	8068,52	?	2,536473	3181
1° do Grupo 19	453	7986,547	?	2,495016	3201
1° do Grupo 20	TFT	7284,825	?	2,273666	3204
1° do Grupo 21	475	7754,236	?	2,449222	3166
1° do Grupo 22	344	8348,337	?	2,721101	3068
1° do Grupo 23	PAVLOV	8186,344	?	2,666562	3070
1° do Grupo 24	524	8940,637	?	2,745053	3257
1° do Grupo 25	545	8590,639	?	2,640836	3253
1° do Grupo 26	TFT	7523,754	?	2,335844	3221
1° do Grupo 27	554	8485,174	?	2,698847	3144
1° do Grupo 28	573	7411,271	?	2,362535	3137
1° do Grupo 29	PAVLOV	7983,366	?	2,518412	3170
1° do Grupo 30	556	7417,502	?	2,347311	3160
1° do Grupo 31	PAVLOV	8725,412	?	2,690537	3243
1° do Grupo 32	PAVLOV	7876,177	?	2,52928	3114
Fase 2					
1° do Grupo 01	TFT	6536,849	?	2,058202	3176
1° do Grupo 02	TFT	7552,689	?	2,364649	3194
1° do Grupo 03	TFT	6754,859	?	2,086765	3237
1° do Grupo 04	TFT	6682,86	?	2,12424	3146
1° do Grupo 05	554	6265,56	?	2,029003	3088
1° do Grupo 06	453	6462,823	?	2,091528	3090
1° do Grupo 07	TFT	6493,743	?	2,063471	3147

1° do Grupo 08	TFT	6987,874	?	2,183028	3201
1° do Grupo 09	TFT	6510,593	?	2,045427	3183
1° do Grupo 10	TFT	6754,384	?	2,146293	3147
1° do Grupo 11	TFT	7243,919	?	2,24757	3223
Fase 3					
1° do Grupo 01	TFT	7116,497	?	2,243536	3172
1° do Grupo 02	555	6187,122	?	1,953622	3167
1° do Grupo 03	TFT	6107,027	?	1,944294	3141
1° do Grupo 04	TFT	6107,626	?	1,921241	3179
1° do Grupo 05	TFT	6458,525	?	2,07004	3120
Fase 4					
1° do Grupo 01	TFT	5372,771	?	1,715992	3131

FONTE: Autor (2019)

Todas as 512 estratégias presentes nesses torneios foram implementadas em Python, o que possibilita a interação com outras estratégias não nebulosos. No torneio original Borges já utilizava três estratégias nebulosas para suas simulações.

4.2.3 Ressimulação do Torneio de Knight

Esse é o torneio mais recente sobre o tema, ele conta com 240 estratégias que disputam todas contra todas, resultando em números próximos de 50 000 iterações a cada simulação. Dos torneios ressimulados esse foi o que demandou o maior custo computacional para execução.

Devido ao tamanho da tabela gerada, iremos apresentar apenas os 15 primeiros colocados da ressimulação, a tabela na integra estará disponível no anexo c deste trabalho.

Tabela 10. Ressimulação Torneio Knight (15 primeiros)

Rank	Estratégia	Ganho Total	Ganho/Adversário	Ganho/Iteração	Iterações
0	Mind Bender	240000	1000	5	48000
1	Hard Prober	158180	659,0833333	3,295416667	48000
2	Handshake	150558	627,325	3,136625	48000
3	ϕ	148856	620,2333333	3,101166667	48000
4	ThueMorse	144000	600	3	48000
5	ThueMorseInverse	144000	600	3	48000
6	Fool Me Forever	143955	599,8125	2,9990625	48000
7	Mirror Mind Reader	143400	597,5	2,9875	48000
8	MoreGrofman	143400	597,5	2,9875	48000
9	More Tideman and Chieruzzi	143400	597,5	2,9875	48000
10	Nice Average Copier	143400	597,5	2,9875	48000
11	N Tit(s) For M Tat(s)	143400	597,5	2,9875	48000

12	Nydegger	143400	597,5	2,9875	48000
13	Omega TFT	143400	597,5	2,9875	48000
14	Once Bitten	143400	597,5	2,9875	48000

FONTE: Autor (2019)

Desde seu trabalho publicado em 2017, que contava com 129 estratégias, já foram adicionadas mais 111 estratégias a biblioteca. Todas as estratégias implementadas nesse trabalho serão submetidas para aprovação e incorporação a biblioteca já existente.

5 CONSIDERAÇÕES FINAIS

5.1 Sumarização e Conclusões

Este trabalho foi proposto com o objetivo de desenvolver uma nova ferramenta que auxiliasse a estudantes e pesquisadores, em seus trabalhos baseados em estratégias de comportamento e na teoria dos jogos.

Como forma de cumprir esse objetivo, foi iniciado o trabalho de coleta, verificação e tradução das estratégias que compõem os torneios computacionais realizados por Axelrod no começo da década de 80, e que estão apresentados nos anexos A e B.

É importante ressaltar que as estratégias do referido torneio foram escritas em Fortran e que não havia um local onde todas estivessem totalmente descritas e organizadas. Esse trabalho aconteceu em paralelo com a divulgação da biblioteca de Vincent Knight.

No primeiro momento acabou sendo gerando um sentimento de frustração, pois o desenvolvimento dessa biblioteca era o objetivo inicial do trabalho. Mas ao analisar o trabalho realizado por Knight, ouve um reforço da necessidade e o potencial de continuar desenvolvendo este tema.

Usando as definições de Russell e Norvig (2013), o trabalho foi reorganizado e como não existia uma ferramenta que atendesse completamente as especificações apresentadas por Cook (2011) e e Kendall, Yao e Chong (2007) surgiu a proposta do desenvolvimento de um protótipo que atendesse essas necessidades.

Surgiu a necessidade de realizar a síntese das estruturas de agentes presentes no trabalho de Russell e Norvig (2013), para acompanhar o processo de classificação das estratégias coletadas. Juntamente com outra necessidade, a de implementar em um só lugar os principais torneios computacionais já realizados.

Nesse momento foi definido o escopo do protótipo, uma ferramenta que pudesse ressimular os principais torneios apresentados por Knight (2017), e que servisse para a criação e estudos das interações entre diversas estratégias. Com o intuito de tornar essa ferramenta ainda mais robusta, inserimos no escopo o torneio

nebuloso de Borges (1996), por se tratar de um marco que criou uma vertente que vem sendo amplamente estudada nos últimos anos.

A implementação das estratégias presentes em Borges (1996), agregaram mais 512 estratégias as 240 já existentes na biblioteca de Vincent Knight. Performando um total de 752 estratégias que podem ser simuladas nos mais diversos modelos de pagamento.

Por fim, a classificação abre uma nova rota de pesquisa, uma vez que particiona as estratégias e pode ser utilizada para obter, de maneira mais eficiente, comportamentos congruentes dos mais variados agentes.

5.2 Contribuições e Trabalhos Futuros

Para facilitar a visualização, as contribuições serão apresentadas em forma de tópicos.

- Agrupamento e descrição das 79 estratégias presentes nos torneios realizados por Axelrod;
- Ferramenta para a ressimulção dos torneios computacionais do Dilema do Prisioneiro;
- Implementação das 512 estratégias do torneio nebulosos de Borges (1996);
- Agrupamento de 752 estratégias das mais variadas, que vão desde reativos simples, até estratégias com aprendizagem de máquina;
- Protótipo de fácil utilização e com uma interface intuitiva, que busca atender a necessidade dos mais variados públicos;
- Classificação das 752 estratégias, segundo a classificação proposta no trabalho;
- Base de dados para estudos futuros usando *machine learning*;

O presente trabalho é apenas o início de uma ferramenta que englobará diversas outras funcionalidades e que vão ser desenvolvidas no decorrer das atividades acadêmicas, tanto em âmbito da graduação como no programa de pós-graduação.

Quanto a evolução do protótipo, foi implementado apenas o torneio nebuloso de Borges (1996), existe a necessidade de implementar os outros torneios nebulosos realizados após esse trabalho.

Outra funcionalidade a ser trabalhada é transformar o protótipo em uma ferramenta para modelagem de agentes, no estilo arrasta-solta, com a funcionalidade de alteração nos parâmetros de ambiente e estruturas dos agentes.

Além de continuar a coleta e implementação das estratégias, buscando aumentar o repositório e as possibilidades de simulação.

6 REFERÊNCIAS

Al-ZANITI, M.; ARAUJO, F.; KUIPER, D.; VALENTE, J.; WENKSTERM, R.Z. DIVAs 4.0: A Multi-Agent Based Simulation Framework. 17th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications. 2013. Disponível em: < https://dl.acm.org/citation.cfm?id=2570883>. Acessado em: 26 de maio 2019.

AMOUROUX, E.; CHU, T.-Q.; BOUCHER, A.; DROGOUL, A. “Gama: An environment for implementing and running spatially explicit multi-agent simulations,” in Proceed-ings of the 10th Pacific Rim International Conference on Multi-Agent Systems (PRIMA 2007), Bangkok, Thai-land, November 21-23, 2007, also in Lecture Notes in.

AQUINO, Jackson Alves . Formação de Alianças e Cooperação entre Antropoides Virtuais: Um Modelo Computacional Baseado em Agentes. Edições UFC. Fortaleza 2012.

AXELROD, Robert. The Evolution of Cooperation. Basic Books, 1984.

BORGES, P. S. S. A Model Of Strategy Games Based on The Paradigm of the Iterated Prisoner’s Dilemma Employing Fuzzy Sets. 1996. Tese (Doutorado em Engenharia de Produção) – Universidade Federal de Santa Catarina, Florianópolis.

BORGES, P. S. S.; PACHECO, Roberto C. S.; BARCIA, Ricardo M.; KHATOR, Suresh. K. **A Fuzzy Approach to the Prisoner's Dilemma**. In Journal of BioSystems, 41, pages 127-137, Elsevier, 1997

BRENNER, W.; ZARNEKOW, R.; WITTIG, H. “Intelligent Software Agents”. Springer, 1998.

CORRÊA, Marcelo França; Vellasco, Marley Maria Bernardes Rebuzzi (Orientadora); Leite, Karla Tereza Figueiredo (Co-orientadora). Modelos NeuroFuzzy Hierárquicos com Aprendizado por Reforço para Multi-Agentes Inteligentes. Rio de Janeiro, 2011. 157p. Tese de Doutorado. Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Crooks, Andrew. The Repast Simulation/Modelling System for Geospatial Simulation. CASA Working Papers . Centre for Advanced Spatial Analysis (UCL), London, UK. 2007.

COOK, CHRIS. Axelrod Tournament, Demonstration Software, 2011. Disponível: http://www2.econ.iastate.edu/tesfatsi/demos/axelrod/axelrodt.htm. Acesso em: 29 maio. 2018

DAMIÃO, M. A.; CAÇADOR, R. M.C.; LIMA, S. M. B.; Princípios E Aspectos Sobre Agentes Inteligentes. Revista Eletrônica da Faculdade Metodista Granbery. N. 17, JUL/DEZ 2014.

FIGUEIREDO, Reginaldo Santana. Teoria dos jogos: conceitos, formalização matemática e aplicação à distribuição de custo conjunto. Gest. Prod., Dez 1994, vol.1, no.3, p.273-289. ISSN 0104-530X

FREITAS, Jackeline Spinola de. **Agentes Inteligentes: benefícios e desafios de sua aplicação na comunicação interativa.** XXV Congresso Brasileiro de Ciências da Comunicação.Salvador/BA. 2002.

HALÁS, Matúš. Game Theoretic Modeling of the International Relations System. 2011. 199 f. Tese (Doutorado em filosofia) - Institute of Political Studies, Charles University, Prague. 2011.

HOMERO. Ilíada. Tradução de Frederico Lourenço. Lisboa: Cotovia, 2005.

HOWE, T. R.; COLLIER N. T.; NORTH, M. J.; PARKER, M. T.; VOS, J. R. "Containing agents: Contexts, projections, and agents," in Proceedings of the Agent 2006 Conference on Social Agents, Chicago, Illinois, 2006, pp. 107–113.

JUCHEM, M., BASTOS, R. M. "Engenharia de Sistemas Multiagentes: Uma Investigação sobre o Estado da Arte", Tecninal Report Series, No. 14, 2001.

KENDALL, G.; YAO, X.; CHONG, S. Y. The Iterated Prisoners' Dilemma 20 Years On ADVANCES IN NATURAL COMPUTATION Series. Vol. 4. 2007.

KNIGHT, Vincent et al. Reinforcement learning produces dominant strategies for the Iterated Prisoner's Dilemma. PLoS ONE, [S.l.], n. 12, p. 1-33, dez. 2017. Disponível em: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0188046>. Acesso em: 01 maio 2018.

KUIPER, D.; WENKSTERN, R. Z. "Virtual agent perception in large scale multi-agent-based simulation systems," in The 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011). Taipei, Taiwan: IFAAMAS, 2011, pp. 1235–1236.

LUKE, S.; CIOFFI-REVILLA, C.; PANAIT, L.; SULLIVAN, K.; BALAN, G. "Mason: A multiagent simulation environment," Simulation, vol. 81, no. 7, pp. 517–527, 2005.

MATHEW, Raj; KAIMAL, M. R. **A Fuzzy Approach to the Prisoner's Dilemma Game Using Fuzzy Expected Value Models**. Proceedings of the Conference on Convergent Technologies for Asia-Pacific Region (TENCON'03), Volume 3, pages 1104-1108, Bangalore, India, 15-17 October 2003.

MÜLLER, J.P. “The design of intelligent agents: a layered approach”. Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, vol. 1177, 1996.

NAKAMITI, Eduardo Kiochi. **Agentes Inteligentes Artificiais.** Dissertação (Mestrado em Comunicação e Semiótica) – Pontifícia Universidade Católica de São Paulo. São Paulo. 2009.

NASH Jr, J. F. Non-Cooperative Games. PhD. Thesis. Princeton University Press, 1950

NORTH, M. J.; HOWE, T. R.; COLLIER, N. T.; VOS, J. R. “The repast simphony runtime system,” in Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms, Chicago, Illinois, 2005, pp. 151–158.

POUNDSTONE, W. Prisoner’s Dilemma: John Von Neumann, Game Theory, and The Puzzle of the Bomb. Anchor Books, 1992.

RAMAN, Kalyan. **A Fuzzy Resolution of the Prisoner's Dilemma**. Technical Reports and Publications of the Center for Study of Complex Systems (CSCS-2002-001). University of Michigan, 2002.

REVISTABW. Inteligência Artificial: Agentes Inteligentes. Revista Brasileira de Web: Tecnologia

REZENDE, S. O. “Sistemas inteligentes: fundamentos e aplicações”. Barueri, SP: Manole. 2005.

RIVEIRO, Sérgio Luiz de Medeiros. Um framework para Simulação Econômica Baseado em um Modelo de Agentes Antecipatórios com Racionalidade Limitada. 1999. 99 p. Dissertação (Mestrado em Engenharia de Produção) - Centro Tecnológico, Universidade Federal de Santa Cararina, Florianópolis, 1999.

RUSSELL, S.; NORVIG, P. Inteligência Artificial: Uma abordagem moderna. 2a. ed. Rio de Janeiro: Elsevier Editora, 2013.

SCHOUERY, R. C. S; LEE, O; MIYAZAWA, F. K.; XAVIER E. C. Tópicos da Teoria dos Jogos em Computação. In: 30º Colóquio Brasileiro de Matemática. Rio de Janeiro: IMPA, 2015.

SILVA, A. P. L.UMA REVISÃO DOS CONCEITOS DA TEORIA DOS JOGOS. Trabalho de Conclusão de Curso (Graduação em Estatística) - Universidade Estadual da Paraíba, Centro de Ciências e Tecnologia, 2014.

SONG, Q.; KANDEL, A. **A Fuzzy Approach to Strategic Games**. IEEE Transactions on Fuzzy Systems, Volume 7, No. 6, December 1999.

STEEL, T.; KUIPER, D.; WENKSTERN, R. “Virtual agent perception in multi-agent based simulation systems,” in Proceedings of IEEE/WIC/ACM International

Conference on Web Intelligence and Intelligent Agent Technology (IAT10), vol. 2. Toronto, Canada: IEEE, 2010, pp. 453–456.

TEIXEIRA, H.; O que é Teoria dos Jogos?. 2015. Disponível em :< http://www.helioteixeira.org/gramatica-da-colaboracao/teoria-dos-jogos-teorias-e-conceitos-chave/>. Acesso em : 05 nov. 2016.

TEIXEIRA, O. N. ALGORITMO GENÉTICO COM INTERAÇÃO SOCIAL NEBULOSA. Tese de Doutorado (Programa de Pós-graduação em Engenharia Elétrica – Computação Aplicada) - Universidade Federal do Pará, Belém, 2012.

von NEUMANN, J. & MORGENSTERN, O. Theory of Games and Economic Behavior. Princeton University Press, Princeton, 1972.

WOOLDRIDGE, M., JENNINGS, N.R. Intelligent agents: Theory and practice. The Knowledge Engineering Review, 10 (2):115-152, 1995.

ANEXOS

ANEXO A PRIMEIRO TORNEIO DE AXELROD

Tabela 11 - Estratégias presentes no primeiro torneio realizado por Axelrod

Posição Final	Nome do Autor	Nome da Estratégia	Linhas de código	Pontuação
01	Anatol Rapoport	TFT	4	504.5
02	Nicholas Tideman & Paula Chieruzzi		41	500.0
03	Rudy Nydegger	NYDEGGER	23	485.5
04	Bernard Grofman		8	481.9
05	Martin Shubik		16	480.7
06	Willian Stein & Ammon Rapoport		50	477.8
07	James W Friedman		13	473.4
08	Morton Davis		6	471.8
09	James Graaskamp		63	400.7
10	Leslie Downing		33	390.6
11	Scott Feld		6	327.6
12	Johann Joss		5	304.4
13	Gordon Tullock		18	300.5
14	Name withheld		77	282.2
15	RANDOM		5	276.3

FONTE: Autor (2019)

Estratégia de Anatol Rapoport (TFT)

Das estratégias submetidas no primeiro torneio essa pode ser considerada uma das mais simples. Na primeira rodada o jogador cooperar e a partir da segunda rodada ele repete o movimento anterior de seu oponente. [1]

Estratégia de Nicholas Tideman & Paula Chieruzzi

Essa estratégia tem uma inicialização semelhante ao TFT [4], a mudança ocorre na resposta às traições. Na primeira traição ocorre a resposta de retaliação simples do TFT, caso a cooperação seja retomada e o oponente volte a trair o jogador retribui com uma traição e incrementa mais uma traição na sequência (n), caso o oponente continue traindo é feita a retribuição normal do TFT, voltando a ser retomada a cooperação e o oponente voltando a trair o jogador retribui com o número de traições anterior incrementando mais uma (n+1) persistindo a traição a resposta e a mesma do TFT. Além do comportamento citado acima existe chama de

“fresh start” ou novo recomeço. Ela ocorre se o oponente tiver uma sequência de duas colaborações consecutivas e estiver com 10 pontos a menos que jogador [6]:

E não comece a trair de novo;

E já tenha se passado 20 rodadas desde a última “fresh start”;

E existe mais de 10 rodadas para terminar o torneio;

E o número de traições tem desvio padrão diferente de 3.0 em uma amostra aleatória 50 – 50.

Estratégia de Rudy Nydegger

Começa com a estratégia TFT para as três primeiras rodadas com exceção se na primeira rodada ele for o único a cooperar e na segunda rodada for o único a trair. Nesse caso na terceira rodada ele irá trair novamente. A partir desse momento as próximas decisões serão baseadas nas três rodadas anteriores seguindo as condições abaixo:

Cada traição do oponente será atribuída um valor de 2;

Cada traição do jogador será atribuída um valor de 1;

O valor resultante da soma entre a pontuação atribuída as traições (n_1) na primeira das três rodadas que estão sendo avaliadas terá o peso 16, o valor (n_2) da segunda rodada um peso 2 e o valor (n_3) da terceira um peso 1, resultando na seguinte fórmula:

$$A = 16n_1 + 4n_2 + n_3$$

Sendo que o jogador apenas trairá se o resultado de A= {1, 6, 7, 17, 22, 23, 26, 29, 30, 31, 33, 38, 39, 45, 49, 54, 55, 58, 61} caso contrário, o jogador irá cooperar.

Estratégia de Bernard Grofman

O jogador inicia cooperando. Caso seu oponente tenha feito o oposto, na próxima roda a uma chance de $\frac{2}{7}$ para cooperação. Caso as ações sejam iguais ele sempre coopera.

Estratégia de Martin Shubik

Essa estratégia tem uma inicialização semelhante ao TFT, a mudança ocorre na resposta as traições. Na primeira traição ocorre a resposta de retaliação simples do TFT, caso a cooperação seja retomada e o oponente volte a trair o jogador retribui com uma traição e incrementa mais uma traição na sequência (n), caso o oponente continue traindo é feita a retribuição normal do TFT, voltando a ser retomada a cooperação e o oponente voltando a trair o jogador retribui com o número de traições anterior incrementando mais uma (n+1) persistindo a traição a resposta e a mesma do TFT.

Estratégia de Willian Stein & Ammon Rapoport

Inicia cooperando durante as 4 primeiras rodadas e depois assume o comportamento TFT, avaliando a cada 15 rodadas pelo teste de qui-quadrado se o oponente está jogando randomicamente. Caso esteja ele só trai.

Estratégia de James W Friedman (Grudger)

Essa estratégia colabora até que seu oponente traía, daí pra frente começa a só trair até o final do jogo.

Estratégia de Morton Davis

Coopera nas 10 primeiras rodadas, caso o oponente traía e começa a trair até o final do jogo.

Estratégia de James Graaskamp

Essa estratégia segue as seguintes regras:

Utiliza a estratégia TFT nas 50 primeiras rodadas;

Na rodada número 51 ele traí;

Nas próximas 5 rodadas ele utiliza TFT;

Depois é feita a verificação se o oponente está jogando de forma aleatória, caso positivo, o jogador começa a trair até o final do jogo;

Caso seja constatado na verificação que o oponente está utilizando a estratégia TFT ou alguma estratégia semelhante o jogador assume TFT até o final do jogo;

Caso contrário ele coopera e traí de forma aleatória a cada 5 ou 15 rodadas.

Estratégia de Leslie Downing

Atualiza duas probabilidades ao longo das jogadas p (cooperações do oponente | cooperações do jogador) e p (cooperações do oponente | traições do

jogador) e seleciona a ação que a longo prazo tem maior expectativa de retorno. Inicialmente as duas probabilidades iniciam em 0,5.

Estratégia de Scott Feld

Inicia utilizando o TFT, é diminui a probabilidade de cooperação após a cooperação do adversário na rodada anterior de maneira que na iteração 200 esse valor é de 0,5.

Estratégia de Johann Joss

Inicia cooperando e joga TFT, com a diferença de que quando o oponente coopera ele tem uma probabilidade de 90% de cooperar.

Estratégia de Gordon Tullock

Coopera nas 11 primeiras rodadas, depois aleatoriamente coopera 10% menos do que o oponente cooperou nas últimas 10 rodadas.

Estratégia Name withheld (Nome omitido)

Essa estratégia coopera baseada em uma probabilidade que inicialmente e igual a 0.3 e é atualizada a cada 10 rodadas baseada se o oponente e aleatório, muito colaborativo ou pouco colaborativo.

Estratégia RANDOM

Coopera ou traí com uma igual probabilidade, ou seja, 0.5.

ANEXO B SEGUNDO TORNEIO DE AXELROD

Tabela 12 - Estratégias presentes no segundo torneio realizado por Axelrod

Posição Final	Nome do Autor	Nome da Estratégia	Linhas de código	Pontuação
01	Anatol Rapoport	TIT FOR TAT	5	434.73
02	Danny C. Champion		16	433.88
03	Otto Borufsen		77	431.77
04	Rob Cave		20	427.76
05	William Adams		22	427.10
06	Jimm Graaskamp & Ken Katzen		23	425.60
07	Heber Weiner		31	425.48
08	Paul D. Harrington	HARRINGTON	112	425.46
09	T. Nicolaus Tideman & P. Chieruzzi		38	425.07
10	Charles Kluepfel		59	425.94
11	Abraham Getzler		9	422.83
12	Francois Leyvraz		29	422.66
13	Edward White, Jr		16	419.67
14	Graham Eatherley		12	418.77
15	Paul E. Black		22	414.11
16	Richard Hufford		45	411.75
17	Brian Yamachi		32	411.59
18	John W, Colbert		63	411.08
19	Fred Mauk		63	410.45
20	Ray Mikkelson		27	410.31
21	Glenn Rowsam		36	410.28
22	Scott Appold		41	408.55
23	Gail Grasell		10	408.11
24	J. Maynard Smith	TIT FOR TWO TATS	9	407.79
25	Tam Almy		142	407.01
26	D. Ambuelh & K. Kickey		23	406.95
27	Craig Feathers	TRANQUILIZER	48	405.90
28	Bernard Grofman		27	403.97
29	Johann Joss	JOSS	74	403.13
30	Jonathan Pinkley		64	402.90
31	Rudy Nydegger	NYDEGGER	23	402.16
32	Robert Pebley		13	400.75
33	Roger Falk & James Langsted		117	400.52
34	Nelson Weiderman		18	399.98
35	Robert Adams		43	399.60
36	Robyn M. Dawes & Mark Batell		29	399.31

37	George Lefevre		10	398.13
38	Stanley F. Quayle		44	397.70
39	R. D. Anderson		44	397.66
40	Leslie Downing	DOWNING	33	397.13
41	George Zimmerman		36	295.33
42	Steve Newman		51	394.02
43	Martyn Jones		152	393.01
44	E. E. H. Shurmann		32	392.54
45	Henry Nussbacher		52	392.41
46	David Gladstein	TESTER	28	390.89
47	Mark F. Batell		30	389.44
48	Devid A. Smith		23	388.92
49	Robert Leyland		52	385.00
50	Michael F. McGurrin		78	383.17
51	Howard R. Hollander		16	380.95
52	James W. Friedman	FRIEDMAN	9	380.49
53	George Hufford		41	344.17
54	Rik Smoody		6	342.89
55	Scott Feld		50	327.64
56	Gene Snodgrass		90	326.94
57	George Duisman		6	309.03
58	W. H. Robertson		54	304.62
59	Horold Rabbie		52	303.52
60	James E. Hall		31	296.89
61	Edward Friedland		84	277.70
62	RANDOM		4	237.22
63	Roger Hotz		14	220.50

FONTE: Autor (2019)

Estratégia de Anatol Rapoport (TFT)

Das estratégias submetidas no primeiro torneio essa pode ser considerada uma das mais simples. Na primeira rodada o jogador cooperar e a partir da segunda rodada ele repete o movimento anterior de seu oponente [6].

Estratégia de Danny C. Champion

Essa estratégia coopera nas primeiras 10 rodadas e joga 15 rodadas utilizando a estratégia TFT. Após essas 25 rodadas ele coopera a menos que: o oponente tenha traído na rodada anterior, e tenha cooperado menos de 60% nas rodadas anteriores e que a taxa de aleatório seja maior que a taxa de traições [1] [3].

Estratégia de Otto Borufsen

Utiliza como estratégia básica o TFT, começando cooperando. Caso aconteça de haver 3 traições mútuas ele começa a cooperar incondicionalmente. A cada 25 rodadas o jogador reavalia se o oponente e o traidor (cooperou menos de 3 vezes na últimas 25 rodadas) ou se o opoente e aleatório (coopera entre 8 e 17 vezes e menos de 70% do que o jogador tenha cooperado), caso identifique que o oponente e traidor ou aleatório começa a trair até o próximo ponto de verificação [1] [3].

Estratégia de Rob Cave

O jogador adota a estratégia de triar após a traição do jogador caso a frequência das traições de seu oponente seja 79, 65, ou 35% de todas as iterações e essa verificação é feita respectivamente nas rodadas 19,29 e 39. Nos outros casos o jogador sempre coopera após a cooperação do adversário, e também caso o oponente traí se ele não tenha traído menos de 18 vezes e o número aleatório entre 0 e 1 seja menor que 0,5 [3].

Estratégia de William Adams

O jogador utiliza a cooperação nos dois primeiros movimentos. Caso o oponente traía menos que o limite estabelecido (inicialmente são 4 traições), o jogador continua cooperando. Depois que esse limiar é atingido o jogador traí e redefine o limite para a meta do anterior, quando esse limite ficar menor do que 1 a decisão é tomada após a comparação do limite e o valor gerado aleatoriamente entre 0 e 1, caso o valor aleatório seja menor que o limite, o jogador coopera [3] [6].

Estratégia de Jimm Graaskamp & Ken Katzen

O jogador começa utilizando TFT, nas rodadas 11ª, 21ª, 31ª, 41ª, 51ª e 101ª o jogador faz a verificação da pontuação. Caso a pontuação entre os confrontos seja até 7 pontos menor que o ideal ele coopera initerruptamente, caso contrário ele começa a trair até o final do jogo [3].

Estratégia de Heber Weiner

O jogador inicia jogando TFT, e muda se o oponente apresentar mais de 4 traições nas 12 rodadas anteriores. Nesse caso o jogador começa a trair independente da última ação do oponente. Caso ocorra uma traição seguida de cooperação e tenha se passado mais de 20 rodadas o jogador esquece essas traições [3] [6].

Estratégia de Paul D. Harrington (revisar código FONTE)

Checa para tentar achar uma vulnerabilidade do oponente. Para isso ele coopera nas 36 primeiras rodadas, na 37ª ele traí para provocar. Caso perceba que o seu oponente está utilizando a mesma estratégia e coopera até o final do jogo. Caso o oponente traía primeiro ele adota comportamento randômico, caso o oponente seja um traidor consistente ele adota o comportamento de traição continua [2].

Estratégia de T. Nicolaus Tideman & P. Chieruzzi

Essa estratégia tem uma inicialização semelhante ao TFT [4], a mudança ocorre na resposta às traições. Na primeira traição ocorre a resposta de retaliação simples do TFT, caso a cooperação seja retomada e o oponente volte a trair o jogador retribui com uma traição e incrementa mais uma traição na sequência (n), caso o oponente continue traindo é feita a retribuição normal do TFT, voltando a ser retomada a cooperação e o oponente voltando a trair o jogador retribui com o número de traições anterior incrementando mais uma (n+1) persistindo a traição a resposta e a mesma do TFT. Além do comportamento citado acima existe chama de “fresh start” ou novo recomeço. Ela ocorre se o oponente tiver uma sequência de duas colaborações consecutivas e estiver com 10 pontos a menos que jogador [6]:

E não comece a trair de novo;

E já tenha se passado 20 rodadas desde a última “fresh start”;

E existe mais de 10 rodadas para terminar o torneio;

E o número de traições tem desvio padrão diferente de 3.0 em uma amostra aleatória 50 – 50.

Estratégia de Charles Kluepfel

Começa cooperado e tem 60% de chance de trair caso o oponente tenha traído na primeira rodada, a partir da segunda rodada o jogador conta o número de cooperações e traições após suas próprias cooperações e traições. Na 26ª rodada o jogador traí em duas hipóteses:

O número de cooperações do oponente após a primeira traição do jogador seja menor que a metade da diferença entre o número de vezes em que o jogador traído primeiro e 3/2 da raiz quadrada desse número;

Em todos os outros casos ele repete o movimento do oponente com probabilidade de 100%, 90%, 70% e 60% se o oponente fizer a mesma coisa nas 3, 2 ou 1 rodadas respectivamente [3].

Estratégia de Abraham Getzler

Inicia cooperando, caso o oponente traía ele tem uma "vida-extra". Basicamente uma função de esquecimento. O jogador só traí se o desconto da soma das traições do outro jogador for maior que o número aleatório entre 0 e 1 [3].

Estratégia de Francois Leyvraz

Essa estratégia baseia seu comportamento nas 3 jogadas anteriores do seu oponente. Para tal ele inicia cooperando e traí com uma probabilidade de 75% caso seu oponente tenha traído nas 3 rodadas ou nas últimas duas. Caso o oponente traía apenas na última rodada, o jogador traí com uma probabilidade de 50%. Em todos os outros casos ele sempre coopera [3].

Estratégia de Edward White, Jr

Coopera nas 10 primeiras rodadas e traí após as traições do oponente, com a condição de o número de traições do oponente multiplicado pelo ln das iterações até o momento seja maior ou igual ao número de iterações [3].

Estratégia de Graham Eatherley

Começa cooperando e traí caso o oponente traía, mais essa traição ocorre com uma probabilidade igual ao número de traições do oponente dividido pelo número de iterações até o momento [3].

Estratégia de Paul E. Black

Coopera nas 5 primeiras rodadas. O jogador continua cooperando caso o número n gerado aleatoriamente multiplicado por 25 e maior que o número de traições elevado a segunda potência menos um [3].

Estratégia de Richard Hufford

Inicia cooperando e joga TFT, caso o oponente tenha feito no último movimento o mesmo que o jogador fez no penúltimo. Ele começa a atualizar o parâmetro de curto prazo (inicialmente 5) é o de longo prazo (inicia igual ao número de encontros até o momento. Se a sensibilidade de longo prazo for acima de nove décimos do número de encontros e simultaneamente a sensibilidade de curto prazo seja igual ao seu valor máximo (5), o jogador coopera. Caso a sensibilidade de longo prazo seja inferior cinco oitavos do total de encontros ou se o valor de curto prazo seja menor que 3, o ator traí. Além disso existe um limiar de RF (inicialmente 20)

Estratégia de Brian Yamauchi

O jogador que utiliza essa estratégia, realiza seus movimentos baseados nas células de uma matriz 2x2. Se o número dor positivo ele coopera, caso não ele traí. Inicialmente todas as células têm valor zero, e o jogo começa na célula superior à esquerda, se o oponente coopera ou traí na rodada anterior e somado ou subtraído um respectivamente do valor da célula. O jogador permanece ou se desloca de células baseado em duas regras:

Após a traição do adversário, a nova célula escolhida será a de baixo;

Caso o jogador decida cooperar, na rodada seguinte vai levar em consideração a célula a esquerda.

Caso o número de encontros seja superior a 40 e a diferença entre cooperação e deserção e inferior a um decimo de todos os movimentos, o jogador opta por trair.

Estratégia de John W, Colbert

Coopera nos 8 primeiros movimentos com exceção do sexto. A partir do 9º movimento o jogador coopera até uma traição do oponente, traí duas vezes seguidas e logo após coopera duas vezes seguidas indiferente da jogada de seu oponente. Segue com esse comportamento após as traições do oponente até o final do jogo.

Estratégia de Fred Mauk

O jogador inicia com a estratégia TFT até o movimento de número 50, no movimento 51 ele traí. Em seguida ele joga TFT por mais 5 rodadas. De acordo com essas 5 rodadas o jogador escolhe entre uma das 4 estratégias disponíveis.

Se o oponente parecer estar utilizando TFT ou a mesma estratégia que o jogador. Nesse caso o jogador assume TFT até o final do jogo;

Caso os ganhos do oponente nas primeiras 56 iterações não sejam maiores que 135, o jogador assume o comportamento de sempre trair;

Se o oponente trair apenas uma vez entre os encontros 51 a 55 e for na rodada número 53, o jogador assume a cooperação até a rodada 118 e depois começa a jogar TFT;

Nos outros casos, o jogador utiliza a estratégia TFT.

Estratégia de Ray Mikkelson

O jogador inicia cooperando nas duas primeiras rodadas, a partir do terceiro encontro o jogador decide através de um parâmetro especial denominado X, que é atualizado a partir da primeira rodada. Inicialmente o valor de X é -3, caso o oponente coopere esse valor sofre um decréscimo de 1 e se o oponente trair e soma 2 ao valor de X. Na terceira rodada o jogador coopera caso o valor de X seja inferior a 3. Caso o número de encontros não tenha ultrapassado 11, ele traí e o valor de X recebe -1. Caso contrário, ou seja, se o número de encontros for maior que 10, o

jogador coopera caso o oponente tenha traído menos de 15% de todos os encontros. Caso contrário ele traí.

Estratégia de Glenn Rowsam

O jogador que utiliza esse tipo de estratégia toma suas decisões baseado em dois parâmetros (KAM e NPHA) que são atualizados a cada rodada e tem valor inicial igual a zero. Se o valor de KAM for maior do que 6, o jogador traí. NPHA é diminuído por 1, cada vez que que receber um valor acima de zero na rodada anterior. Além disso, a menos que NPHA seja igual a 1 antes dessa redução o jogador coopera. Se nenhuma dessas condições for atendida, o valor de KAM e diminuído a cada 18 encontros desde que ele seja maior que 2. Diante de tudo isso o jogador coopera com exceção de a cada 6 encontros, quando se considera a regra a seguir. Se a pontuação obtida até agora é, pelo menos 2,5 vezes maior (ou igual) o número de encontros multiplicado pelo valor de pagamento a punição, então o jogador coopera. Caso contrário, o jogador traí e o valor de KAM é aumentado em 1. Além disso, se a pontuação adquirida até agora é menor que 2x , 1,5x ou 1x o número de encontros vezes o pagamento por punição, então em vez de KAM ser incrementado por 1, ele é incrementado por 2,3 ou 5 respectivamente (a opção de traição é retirada).

Estratégia de Scott Appold

O jogador utiliza dos parâmetros chaves que são atualizados a cada rodada. O primeiro parâmetro e a razão do número de traições do oponente após a colaboração do jogador sobre o número de colaborações do jogador (inicialmente igual a 1). O segundo parâmetro representa a proporção do número de traições do oponente após o jogador ter traído também sobre o total de traições do jogador. O jogador coopera nos quatro primeiros encontros e fica observando quando o jogador começa a trair, após a traição caso o jogador tenha colaborado no penúltimo encontro e também o primeiro parâmetro e maior do que o número gerado aleatoriamente ele traí. Do mesmo modo se o jogador traiu no penúltimo encontro e simultaneamente o segundo parâmetro é maior do que o número gerado aleatoriamente ele também traí.

Estratégia de Gail Grasell

O jogador começa cooperando e continua até que o valor de traições do oponente seja maior que a metade do número total de iterações.

Estratégia de J. Maynard Smith

Inicia cooperando e continua até que o outro jogador traía duas vezes consecutivas. A cooperação é restaurada assim que o outro jogador volte a cooperar.

Estratégia de Tam Almy

Começa cooperando e depois escolhe uma das quatro estratégias disponíveis (TFT, TFTT, ALLD, explorar estratégia). Quando a estratégia é escolhida ela é utilizada por 10 rodadas seguida de uma avaliação e eventual mudança, e redefinição da contagem de traições tanto do oponente quanto das suas. A estratégia de exploração é escolhida caso o oponente ainda não tenha desertado, ou se ela já havia sido selecionada antes, ou se o jogador também não traiu nos últimos 10 encontros (Para lógica detalhada de exploração ver código FONTE). Em circunstâncias especiais, quando a estratégia selecionada apresentar desempenho pior do que a última estratégia selecionada e houver pelo menos uma traição mútua nos 10 últimos encontros, na sequência o jogador coopera e prossegue novamente para a avaliação (estratégia exploratória é descartada).

Estratégia de D. Ambuelh & K. Kickey

O jogador inicia cooperando e repete o mesmo movimento do oponente nos próximos quatro encontros e depois coopera caso o oponente tenha cooperado na maioria dos 5 encontros anteriores.

Estratégia de Craig Feathers

O jogador começa cooperando e conta o número de traições do oponente desde sua última cooperação (S), assim como o número total de cooperações até o momento (C). Se a soma dos ganhos com as iterações anteriores é de pelo menos ¾ dos ganhos possíveis de cooperação mútua da matriz inicial de pagamento e se o número aleatório de 0 a 1 não é maior que P, o jogador traí uma rodada e depois

coopera em duas. P é igual a 0,95 acrescentado por um sobre o número de encontros elevado ao quadrado, subtraído pela soma da média de todos os pagamentos resultantes das duas cooperações incondicionais após a traição mais 5 e dividido por 15, e menos 0,25 caso o oponente tenha traído no encontro anterior. No entanto, caso a soma dos ganhos seja menor que ¾, em seguida, o jogador traí a menos que duas condições se cumpram.

Primeira: se a soma geral dos pagamentos é ainda menor que 7/12 dos ganhos possíveis de cooperação mútua, em seguida, o jogador repete o movimento anterior do adversário;

Segunda: se o número aleatório de 0 a 1 é no máximo ¼ acrescido de (C) pela diferença de ganhos entre o jogador e seu oponente nas suas iterações anteriores dividido por 100, e por 4 vezes um sobre o número de iterações, menos ¼ do valor de (S). Então o jogador coopera.

Estratégia de Bernard Grofman

O jogador coopera nos dois primeiros movimentos e depois utiliza TFT nos próximos 5 movimentos. A partir do 8 movimento o jogador coopera se ele cooperou na última rodada e seu oponente traiu menos de 3 vezes nas 7 rodadas anteriores, ou se o jogador traiu na última rodada e o oponente traiu no máximo uma vez nas últimas 7 rodadas. Caso contrário ele traí.

Estratégia de Johann Joss

O jogador inicia cooperando e, posteriormente, conta as cooperações de seu oponente e decide seu comportamento a partir de 5 estágios diferentes, onde cada um atribui um estado diferente (estados de 1 a 5, iniciando em 1). No estágio 1, o jogador traí com uma probabilidade de 10%, juntamente com a mudança de estado para 5, caso não ocorra traição, o jogador procede da mesma maneira, exceto para o primeiro passo do 5 estágio que é mudar o estágio para 4. Baseado nos passos de 1 e 5, o jogador primeiro redefine o número de traições do oponente caso ele tenha cooperado na rodada anterior, ou então, aumenta o número de traições. Se esse número for superior a 20 cooperações, o jogador muda para o estado 3 e reinicia as traições. Em outros casos (as traições ou cooperações do adversário não excedem

20), o jogador verifica se o oponente cooperou em pelo menos 70% das rodadas desde o início até o penúltimo encontro. Se for assim o jogador repete a última jogada do adversário. Se não ele muda para o estado 2, nesse estado o jogador traí, atualiza o número de traições do oponente se o adversário traiu na rodada anteior, e caso esse número seja maior do que 10 ele muda para o estado 3, caso não ele se mantém no estado 2 e realiza a atualização dos parâmetros do oponente. No estado 3 o jogador atualiza o número de traições do oponente e verifica se é superior a 20, caso seja ele coopera e redefine esse número sem alterar o estado. Em outro caso ele repete a última jogada do adversário. No estado 4, o jogador sempre coopera. Ele muda para o estado um, se o adversário na rodada anterior, e caso contrário ele aumenta do grau de generosidade em um (inicialmente 0). Quando esse parâmetro atinge o valor de 4, o jogador redefine o número de traições e muda para o estado 3. Caso contrário o jogador continua de acordo com o estado 1 na próxima rodada.

Estratégia de Jonathan Pinkley

Coopera nos dois primeiros movimentos e depois analisa o comportamento do oponente através de um único passo do processo de MARKOV, atualizando a probabilidade de o oponente colaborar depois dos quatro possíveis estados finais (CC, DD, CD, DC) no último encontro.

Estratégia de Rudy Nydegger

Começa com a estratégia TFT para as três primeiras rodadas com exceção se na primeira rodada ele for o único a cooperar e na segunda rodada for o único a trair. Nesse caso na terceira rodada ele irá trair novamente. A partir desse momento as próximas decisões serão baseadas nas três rodadas anteriores seguindo as condições abaixo:

Cada traição do oponente será atribuída um valor de 2;

Cada traição do jogador será atribuída um valor de 1;

O valor resultante da soma entre a pontuação atribuída as traições (n_1) na primeira das três rodadas que estão sendo avaliadas terá o peso 16, o valor (n_2) da segunda

rodada um peso 2 e o valor (n_3) da terceira um peso 1, resultando na seguinte fórmula:

$$A = 16n_1 + 4n_2 + n_3$$

Sendo que o jogador apenas trairá se o resultado de A= {1, 6, 7, 17, 22, 23, 26, 29, 30, 31, 33, 38, 39, 45, 49, 54, 55, 58, 61} caso contrário, o jogador irá cooperar.

Estratégia de Robert Pebley

Começa cooperando, e repete a ação do oponente caso eles tenham tido o mesmo comportamento na rodada anterior, caso não, ele traí com uma probabilidade de 80%.

Estratégia de Roger Falk & James Langsted

O jogador inicia cooperando. Depois ele verifica o resultado das 8 primeiras iterações e conta quantas vezes o oponente cooperou/traiu após as suas próprias cooperações e traições. Caso o oponente tenha traído nas últimas 8 rodadas, o jogador assume o comportamento TFT, a menos que perceba que o jogador é do tipo aleatório. Em seguida ele traí e redefine o parâmetro de traição (D). O adversário é considerado aleatório se o valor de suas cooperações com as taxas de cooperação e traição do jogador são entre 3:2 e 1:2. Depois checa se o oponente esta em modo TFT. Em seguida, se o parâmetro (C) de cooperação é positivo, ele coopera e redefine esse parâmetro para zero. Depois ele avalia se o usuário tem apenas cooperado após suas cooperações. Se for verdade, ele coopera. Se não, ele verifica se o oponente respondeu a uma única cooperação nas 4 rodadas anteriores de acordo com a logica TFT. Se sim, o jogador coopera e define o parâmetro (C) para um. Caso não o jogador traí e reajusta o valor de (D) se o oponente traiu em pelo menos 3 rodadas com frequência. O jogador redefine o parâmetro (D) se qualquer uma das últimas 5 condições é considerada verdadeira, exceto a terceira. Ele coopera caso as condições 2, 3 e 5 forem verdadeiras e a última for falsa. Essas condições são avaliadas na seguinte ordem:

Se o jogador foi o único que traiu na rodada anterior;

O parâmetro (D) é positivo;

Se ambos cooperaram na rodada anterior;

Se o jogador colaborou em apenas uma rodada anterior;

Ou se ele traiu.

Estratégia de Nelson Weiderman

Começa cooperando e continua cooperando até que seu oponente traí 3 vezes consecutivas, daí pra frente ele começa a trair até o final do jogo.

Estratégia de Robert Adams

O jogador inicia cooperando e utiliza TFT até a segunda traição do oponente. No entanto, se a primeira traição ocorrer na primeira rodada o jogador não traí nas rodas de número 3, 6, 9 e assim por diante. Da mesma forma, caso a primeira traição ocorra na segunda rodada ou nas rodadas posteriores, o jogador não retalia e também não traí nas rodadas de número 4,7,10 etc. Mas, essa condição de não retaliar a traição do oponente pode ser impedida (por uma única rodada) pela probabilidade de cooperar (inicialmente 80%), está probabilidade é reduzida pela metade após cada traição do oponente, que é antecedida por uma cooperação. Em todos os outros casos o jogador coopera.

Estratégia de Robyn M. Dawes & Mark Batell

O jogador coopera na primeira rodada e continua cooperando caso o oponente coopere. Caso o oponente traía o jogador só coopera se o produto de 1,6667 elevado a soma de todas as traições do oponente por 0,882 elevado a soma das cooperações for maior que 5. Caso contrário o jogador começa a trair até o final do jogo.

Estratégia de George Lefevre

O jogador utilizando essa estratégia só traí se as traições do adversário ultrapassarem um quinto de todas as iterações já realizadas.

Estratégia de Stanley F. Quayle

Inicia cooperando nas duas primeiras rodadas e depois começa a decidir baseado nas respostas do seu oponente. Para isso ele atualiza dois parâmetros “bom” e “mau”. Caso o adversário coopere após o jogador cooperar essa cooperação fica armazenada no parâmetro “bom”, caso ele traía fica armazenado no parâmetro “mau”. Além desses dois parâmetros existem mais dois “c” e “alt” que são definidos por:

$$c = 6 * bom - 8 * mau - 2$$
$$alt = 4 * bom - 5 * mau - 1$$

Se “c” não é negativo e “alt” é inferior ou igual a “c” o jogador coopera. Caso a primeira situação não ocorra e a segunda sim o jogador faz o movimento oposto ao feito na rodada anterior. Fora isso todas as outras situações causam a traição.

Estratégia de R. D. Anderson

Começa cooperando nos dois primeiros movimentos. Após o segundo encontro conta quantas vezes o adversário respondeu cooperação com cooperação e cooperação com traição. Nas 15 primeiras iterações o jogador traí a menos que o oponente coopere no encontro anterior ou se o número de traições do oponente é maior que 2.

A partir da 16ª iteração o jogador traí se o número de traições do oponente é igual a número de vezes que o jogador cooperou primeiro ou superior a 1/3 da soma de todas as vezes que o jogador cooperou primeiro. Se não for o caso, o jogador escolhe cooperar em todas as rodadas exceto a cada quarto do jogo, e mesmo a cada quarto do jogo se o adversário traiu apenas uma vez nos 16 encontros o jogador coopera. Se não for o caso, o jogador ao trair a primeira vez não recebeu uma traição de retaliação do oponente, ou se a soma das cooperações do oponente após a traição do jogador é pelo menos igual a 1/12 de todas as iterações ocorridas até o momento. Em todas as outras possibilidades o jogador coopera.

Estratégia de Leslie Downing

Essa estratégia tenta estimar o próximo movimento do oponente baseado em probabilidade de cooperação do oponente e sua traição p=(C|D) e na probabilidade de cooperação do oponente e na sua também p=(C|C). Essa probabilidade inicia igual, ou seja, 0.5 para cada.

Estratégia de George Zimmerman

O jogador inicia cooperando e caso seu oponente também faça a mesma jogada ele continua cooperando. No entanto caso o adversário tenha optado por traição, o jogador conta quantas vezes ele traiu/cooperou sozinho. Ele continua na opção escolhida até o número cooperações/traições unilaterais do seu oponente chegar a um determinado limiar (respectivamente 4/8). Então o jogador adota comportamento alternado do que ele vinha utilizando até o momento e atualiza os limites e reinicia a contagem de cooperações/traições unilaterais do seu oponente. Se o jogador cooperou até o momento dele atualizar o valor do limite de cooperação, esse valor é o número de traições unilaterais do oponente incrementado de 1 e multiplicado por 1,6667. Caso o jogador tenha traído até o momento ele atualiza o valor do limite de traições unilaterais, esse valor é o valor antigo menos 3 e somado ao valor inteiro da pontuação adquirida nos encontros anteriores dividido pelo produto das recompensas e punições nos encontros.

Estratégia de Steve Newman

O jogador inicia cooperando e repete esse comportamento nos dois primeiros movimentos, e em seguida atualiza os valores de BETA e ALPHA de cooperação do oponente após traição/cooperação do jogador. Para decidir o que vai fazer o jogador constrói dois parâmetros (A e B), o valor do parâmetro “A” é (6*ALPHA-2) + 9*BETA e o valor de “B” é (4*ALPHA-1) + 6*BETA. Caso “A” não seja negativo e inferior a “B” então o jogador coopera. Caso a seja o único valor positivo, o jogador tem o comportamento inverso ao que teve na rodada anterior. Ele faz o mesmo se “A” é menor que zero e “B” é positivo. No entanto caso “A” e “B” sejam negativos o jogador traí nas três primeiras situações semelhantes, caso contrário ele coopera e reseta o número de casos que isso ocorreu caso ou ele ou o oponente tenham cooperado na rodada anterior.

Estratégia de Martyn Jones

Logica complicada!

Estratégia de E. E. H. Shurmann

Inicia cooperando e continua cooperando desde que o oponente faça o mesmo, caso o oponente traía pelo menos uma vez o jogador define seu movimento a partir do parâmetro (inicialmente 0,5) que é atualizado a cada encontro. Caso haja cooperação mútua na rodada anterior o jogador coopera com um valor do parâmetro vezes 0,57 mais 0,43. Em caso de traição mútua o jogador coopera com o valor do parâmetro multiplicado por 0,74 mais 0,104. Caso apenas um dos jogadores tenha traído, ele colabora com o parâmetro igual a 0,5.

Estratégia de Henry Nussbacher

O jogador coopera nos 10 primeiros movimentos. O jogador conta o número de traições do oponente nesses 10 movimentos e decide seus próximos movimentos baseados os valores obtidos. Se o oponente desertou 9 ou 10 encontros o jogador traí com uma probabilidade de 94%. Caso o oponente tenha traído 7,6,5 ou 2 vezes seguidas, o jogador traí com uma probabilidade de 87%. No caso de 4,3 ou 8 traições do outro jogador, o ator traí com uma probabilidade de 91,5%. Se o oponente traiu apenas uma vez o jogador traí com uma probabilidade de 23 %. Caso o oponente colabore de forma consistente o jogador também coopera de igual modo.

Estratégia de David Gladstein

O jogador começa traíndo. Após a primeira traição do adversário ele coopera e começa a usar a estratégia TFT. Entre a primeira rodada e a primeira traição do oponente o jogador traí se o número cooperações em relação as cooperações do outro jogador somada por um é igual ou superior que a metade. Assim ele traí no 4,6, 8... encontros.

Estratégia de Mark F. Batell

Começa cooperando, e traí acaso a traição do seu adversário seja separada por menos de 3 cooperações. Nesse caso ou no caso de o oponente ultrapassar a décima traição, o jogador traí até o final do jogo.

Estratégia de Devid A. Smith

Coopera no primeiro movimento com uma probabilidade de 0,95, e também após cooperação do adversário e após traição do oponente precedida por cooperação. O jogador coopera com uma probabilidade de 0,05 após 2,3,4 ou 5 traições seguidas. Após a 6ª traição consecutiva o número de traições é zerado e o jogador volta a cooperar com probabilidade de 0,95. Caso continue com as traições ininterruptas o jogador retarda suas tentativas de cooperação para a 6ª,14 ª,25 ª,49 ª, ... iterações.

Estratégia de Robert Leyland

O jogador utilizando essa estratégia inicia com a cooperação e atualiza a cada rodada o número total de traições do oponente, assim como o número de traições após a última cooperação do oponente (I5). O parâmetro I5 é reiniciado após cada cooperação do oponente, exceto se esse valor for inferior a 2. Esse parâmetro também é reiniciado após toda a cooperação do jogador que não é baseada no TFT, por exemplo, quando o jogador repete a cooperação do oponente dado I5 maior do que 5, o que é sempre a primeira condição avaliada no processo de tomada de decisão após atualizar o número de traições. Em outra situação o jogador utiliza TFT, caso o número de iterações é inferior a 30. Se o número de encontros ultrapassar 29, o jogador reconsidera a cooperação com uma probabilidade inicial de 75% e diminui por 20 se o oponente traiu de 40 a 60% de todas as iterações anteriores. Do encontro 30 em diante o jogador geralmente repete o último movimento do oponente a menos que o número aleatório entre 0 e 1 seja maior que a probabilidade cooperação, nesse caso o jogador traí. Caso aconteça a traição baseada na condição anterior, o jogador caso se considere em uma posição “ruim” e ele volta a uma posição “boa” apenas depois de cooperar não baseado em TFT, da mesma forma da contagem da recente traição. Depois de cada traição causada exclusivamente pela baixa probabilidade de cooperação, o jogador repete o último movimento do oponente (se não trair por causa da baixa probabilidade de cooperação) e, em seguida, até restaurar a “boa” posição, ele toma conhecimento do último movimento do oponente. Se o oponente cooperar, o jogador diminui a probabilidade de cooperar em 5 pontos, se o resultado for negativo redefine o valor para zero e repete a última ação do oponente, se o resultado é de pelo menos 0,3 ,

ou se não for o caso ele prossegue para a obtenção do número aleatório e comparação com a probabilidade de cooperação. Mas se o adversário traiu, e o jogador cooperou, aumenta a probabilidade de cooperação em 15 pontos, não podendo ultrapassar o valor 1 e garante que o valor de I5 é maior que 5, ele coopera na próxima rodada (mesmo sem ter atualizado a probabilidade de cooperação) e repete até que I5 não seja mais inferior a 6. Neste momento as regras para redefinir a contagem de traições recentes já foi iniciada.

Estratégia de Michael F. McGurrin

Inicia traindo e nas duas rodadas seguintes ele coopera, ele analisa os movimentos do seu oponente. Caso o adversário tenha traído na primeira e segunda rodada ele altera seu comportamento para TFT. Caso o jogador tenha cooperado nos dois primeiros movimentos o jogador introduz um traição a cada 8 rodadas. E cooperando caso o oponente tenha cooperado em pelo menos um dos dois movimentos anteriores. Caso o jogador tenha cooperado em apenas uma vez nas duas primeiras rodadas o jogador utiliza o comportamento desse jogador na terceira rodada para definir seu comportamento. Caso ele coopere na 3ª rodada o jogador altera seu comportamento para TF2T, caso o adversário tenha traído no 3ª rodada muda sua estratégia para TFT, iniciando a 4ª rodada com o movimento que seu oponente usou no primeiro movimento.

Estratégia de Howard R. Hollander

O jogador que utiliza essa estratégia apenas traí caso seu oponente traía duas vezes consecutivas. Além disso ele introduz traições aleatórias que vão diminuindo de frequência no decorrer do jogo.

Estratégia de James W. Friedman

Essa estratégia colabora até que seu oponente traía, daí pra frente começa a só trair até o final do jogo.

Estratégia de George Hufford

O jogador usa TFT nos 5 primeiros encontros e salva o número de traições nestes movimentos. Se as últimas 5 rodadas forem tão lucrativas quanto as 5

primeiras e se o número de traições for menor que 5 o jogador introduz uma traição (cada 5 movimentos). Caso essa decisão traga ganhos inferiores os 5 movimentos anteriores o jogador reduz o número de traições, caso os ganhos continuem abaixo do esperado, o jogador reinicia a estratégia.

Estratégia de Rik Smoody

Coopera e apenas traí com uma probabilidade de 10%, caso o seu oponente coopere na rodada anterior.

Estratégia de Scott Feld

O jogador que utiliza esse tipo de estratégia escolhe a cada 20 rodadas seu padrão de comportamento. Existem 5 padrões básicos de comportamento:

Sempre cooperar;

Cooperar, mesmo após uma traição com probabilidade de 25%;

Jogar TFT;

Trair após uma cooperação com probabilidade de 25%;

Sempre trair;

O jogador joga TFT nas primeiras 20 iterações e gradualmente se move para o padrão de sempre trair. O jogador permanece nesse padrão a menos que seus ganhos sejam inferiores ao padrão cooperativo mais próximo (últimos 20 movimentos quando estava usando o padrão 4). Se esse padrão de menos traição assegurar maior pontuação (mais não tão alto como o padrão mais cooperativo), então ele se move gradualmente para o padrão mais cooperativo. Todas as mudanças para o padrão mais cooperativo são acompanhada por suposição de que o adversário colaborou na última rodada. Depois de atingir o padrão 1, ele permanece nele a menos que tenha um ganho menor do que o último comportamento menos cooperativo. Nesse caso ele se move novamente para a extremidade de sempre trair e emprega condições semelhantes a que ele utilizou para subir de traição para cooperação: ele continua diminuindo a probabilidade à

medida que os ganhos no próximo padrão são promissores, caso não sejam e o padrão de cooperação anterior também não ele permanece no padrão atual.

Estratégia de Gene Snodgrass

Inicia cooperando nas 10 primeiras rodadas, depois por 10 rodadas ele traí e ele começa a alternar entre cooperação e traições nas próximas 10 rodadas, depois joga TFT por mais 10 rodadas e finaliza a sequência com 10 rodadas utilizando a regra TF2T. Essas 5 estratégias são alternadas a cada 10 rodadas. Após a 10ª rodada é iniciada a avaliação de ganhos, essa avalição é feita a cada 10 rodadas, caso a média dos pontos seja inferior à média anterior a última estratégia é desativada do ciclo e só volta a ser ativada caso o ganho com ela seja maior que 90 % do que as médias das estratégias ativas.

Estratégia de George Duisman

O jogador coopera em todas as rodadas ímpares.

Estratégia de W. H. Robertson

O jogador inicia cooperando e repete o movimento anterior do até a 4ª rodada. Após a 4ª rodada o jogador começa a contar o número total de traições do adversário, número de traições consecutivas desde a última cooperação e o número de cooperações consecutivas do adversário. Após a cooperação do adversário o jogador coopera a menos que o número de traições do adversário seja maior que o limite estabelecido (inicialmente 20% de todas as iterações). Mesmo nesse caso o jogador só coopera caso o número total de traições é pelo menos 20 vezes menor que o número de cooperações consecutivas vezes o número total de encontros. Além disso, o jogador traí após cooperação do oponente se o número da rodada e divisível por 12. Esse número é reduzido por um a cada 6 traições introduzidas desse modo. Esse processo de traição por divisibilidade e abandonado caso o oponente tenha respondido com traição após essa introdução de traição. Depois de traição do adversário o jogador coopera caso o número de traições não seja maior que o limite de traições ou caso o jogador tenha traído em 3 rodadas seguidas. Após 20 rodadas o jogador diminui o limite de traições para 10% de todas as iterações.

Estratégia de Horold Rabbie

O jogador inicia cooperando e faz isso para os próximos 20 movimentos, a menos que o índice calculado a cada rodada seja maior do que 2. Esse índice é calculado a partir da segunda rodada em diante e recebe os valores de 4,3,2, e 1 respectivamente se o jogador fizer nas duas rodadas anteriores o seguinte:

Traiu em ambos os encontros;

Ele traiu apenas na penúltima rodada;

Traiu no último movimento;

Não traiu;

Caso o oponente se mostre um copiador de estratégia consistente, o jogador a partir do 23º encontro começa a cooperar até que seu oponente escolha trair. Após o 22ª encontro o jogador usa estratégias diferentes para interagir, ele escolhe uma entre as 6 estratégias disponíveis. A estratégia que o jogador irá utilizar depende da probabilidade de cooperação do adversário após as 4 possíveis combinações de movimento do jogador nos últimos 2 encontros, dado por pesos diferentes a essa probabilidade dentro dessas 6 estratégias.

Estratégia de James E. Hall

Coopera no primeiro movimento e só coopera no próximo caso o oponente tenha traído no anterior. Depois disso ele coopera nos casos de cooperação e traição ímpares do adversário. Nos outros casos ele traí.

Estratégia de Edward Friedland

Inicia cooperando e depois calcula a probabilidade de cooperação do adversário após suas cooperações (ALPHA) ou traições (BETA). Caso o oponente seja identificado como jogador aleatório e adotada a estratégia de traição contínua. O oponente e considerado aleatório se faz:

Se ele usa a mesma estratégia por três rodadas seguidas;

Se não escolhe o mesmo que foi escolhido por 10 rodadas consecutivas;

Se ele desertou 10 a 26 excluindo as 36 iterações;

E se ele mudou de escolha menos de 26 vezes;

Caso o jogador não seja considerado com o jogador escolhe uma das 3 estratégias baseada nos valores de ALPHA e BETA: sempre cooperar, sempre trair e alternar entre trair e cooperar.

Estratégia de RANDOM

Coopera ou traí com uma igual probabilidade, ou seja, 0.5.

Estratégia de Roger Hotz

O jogador coopera com uma probabilidade de 0,10 nos primeiros 100 movimentos, nos próximos 100 essa probabilidade é de 0,05, nos outros 100 é de 0,15 .

ANEXO C RESULTADO DA RESSIMULAÇÃO DO TORNEIO DE KNIGHT

Tabela 13 - Resultados completos da ressimulação dos torneios de Knight

Rank	Estratégia	Ganho Total	Ganho/Adversário	Ganho/Iteração	Iterações
0	Mind Bender	240000	1000	5	48000
1	Hard Prober	158180	659,0833333	3,295416667	48000
2	Handshake	150558	627,325	3,136625	48000
3	ϕ	148856	620,2333333	3,101166667	48000
4	ThueMorse	144000	600	3	48000
5	ThueMorseInverse	144000	600	3	48000
6	Fool Me Forever	143955	599,8125	2,9990625	48000
7	Mirror Mind Reader	143400	597,5	2,9875	48000
8	MoreGrofman	143400	597,5	2,9875	48000
9	More Tideman and Chieruzzi	143400	597,5	2,9875	48000
10	Nice Average Copier	143400	597,5	2,9875	48000
11	N Tit(s) For M Tat(s)	143400	597,5	2,9875	48000
12	Nydegger	143400	597,5	2,9875	48000
13	Omega TFT	143400	597,5	2,9875	48000
14	Once Bitten	143400	597,5	2,9875	48000
15	PSO Gambler 1_1_1	143400	597,5	2,9875	48000
16	PSO Gambler 2_2_2	143400	597,5	2,9875	48000
17	PSO Gambler 2_2_2 Noise 05	143400	597,5	2,9875	48000
18	PSO Gambler Mem1	143400	597,5	2,9875	48000
19	Punisher	143400	597,5	2,9875	48000
20	Random Hunter	143400	597,5	2,9875	48000
21	Resurrection	143400	597,5	2,9875	48000
22	Retaliate	143400	597,5	2,9875	48000
23	Retaliate 2	143400	597,5	2,9875	48000
24	Retaliate 3	143400	597,5	2,9875	48000
25	Revised Downing	143400	597,5	2,9875	48000
26	Rowsam	143400	597,5	2,9875	48000
27	ShortMem	143400	597,5	2,9875	48000
28	Shubik	143400	597,5	2,9875	48000
29	Slow Tit For Two Tats 2	143400	597,5	2,9875	48000
30	Soft Grudger	143400	597,5	2,9875	48000
31	Soft Joss	143400	597,5	2,9875	48000
32	Spiteful Tit For Tat	143400	597,5	2,9875	48000
33	TF3	143400	597,5	2,9875	48000
34	Thumper	143400	597,5	2,9875	48000
35	Tideman and Chieruzzi	143400	597,5	2,9875	48000
36	Tit For Tat	143400	597,5	2,9875	48000
37	Tit For 2 Tats	143400	597,5	2,9875	48000
38	Tricky Level Punisher	143400	597,5	2,9875	48000
39	Two Tits For Tat	143400	597,5	2,9875	48000
40	UsuallyCooperates	143400	597,5	2,9875	48000

41	VeryBad	143400	597,5	2,9875	48000
42	Weiner	143400	597,5	2,9875	48000
43	White	143400	597,5	2,9875	48000
44	Winner12	143400	597,5	2,9875	48000
45	Win-Stay Lose-Shift	143400	597,5	2,9875	48000
46	WmAdams	143400	597,5	2,9875	48000
47	Worse and Worse 3	143400	597,5	2,9875	48000
48	Yamachi	143400	597,5	2,9875	48000
49	ZD-GTFT-2	143400	597,5	2,9875	48000
50	ZD-GEN-2	143400	597,5	2,9875	48000
51	Dynamic Two Tits For Tat	143400	597,5	2,9875	48000
52	Meta Hunter	143400	597,5	2,9875	48000
53	Meta Majority	143400	597,5	2,9875	48000
54	Meta Majority Finite Memory	143400	597,5	2,9875	48000
55	Meta Majority Long Memory	143400	597,5	2,9875	48000
56	NMWE Deterministic	143400	597,5	2,9875	48000
57	NMWE Finite Memory	143400	597,5	2,9875	48000
58	NMWE Long Memory	143400	597,5	2,9875	48000
59	NMWE Memory One	143400	597,5	2,9875	48000
60	NMWE Stochastic	143400	597,5	2,9875	48000
61	Nice Meta Winner	143400	597,5	2,9875	48000
62	Nice Meta Winner Ensemble	143400	597,5	2,9875	48000
63	Opposite Grudger	143397	597,4875	2,9874375	48000
64	Prober 2	143397	597,4875	2,9874375	48000
65	Pun1	143397	597,4875	2,9874375	48000
66	SolutionB5	143397	597,4875	2,9874375	48000
67	Stalker	143397	597,4875	2,9874375	48000
68	Suspicious Tit For Tat	143397	597,4875	2,9874375	48000
69	Willing	143397	597,4875	2,9874375	48000
70	Winner21	143397	597,4875	2,9874375	48000
71	Meta Majority Memory One	143397	597,4875	2,9874375	48000
72	SolutionB1	143394	597,475	2,987375	48000
73	Stein and Rapoport	143394	597,475	2,987375	48000
74	Risky QLearner	143376	597,4	2,987	48000
75	Remorseful Prober	143337	597,2375	2,9861875	48000
76	ZD-Mem2	143334	597,225	2,986125	48000
77	Worse and Worse	143331	597,2125	2,9860625	48000
78	Tullock	143325	597,1875	2,9859375	48000
79	Tranquilizer	143316	597,15	2,98575	48000
80	RichardHufford	143307	597,1125	2,9855625	48000
81	Stochastic Cooperator	143298	597,075	2,985375	48000
82	Meta Mixer	143298	597,075	2,985375	48000
83	Stochastic WSLS	143295	597,0625	2,9853125	48000
84	Random Tit for Tat	143268	596,95	2,98475	48000
85	ZD-Extort-2	143241	596,8375	2,9841875	48000
86	SelfSteem	143235	596,8125	2,9840625	48000
87	ZD-Extort-2 v2	143235	596,8125	2,9840625	48000

88	ZD-SET-2	143205	596,6875	2,9834375	48000
89	ZD-Extort3	143184	596,6	2,983	48000
90	TF2	143160	596,5	2,9825	48000
91	ZD-Extort-4	143136	596,4	2,982	48000
92	ZD-Extortion	143109	596,2875	2,9814375	48000
93	TF1	143103	596,2625	2,9813125	48000
94	Ripoff	143100	596,25	2,98125	48000
95	Tester	143100	596,25	2,98125	48000
96	Win-Shift Lose-Stay	143100	596,25	2,98125	48000
97	Random	143073	596,1375	2,9806875	48000
98	Worse and Worse 2	143007	595,8625	2,9793125	48000
99	ZD-Mischief	142965	595,6875	2,9784375	48000
100	Prober 4	142830	595,125	2,975625	48000
101	Meta Hunter Aggressive	142812	595,05	2,97525	48000
102	Tricky Cooperator	142809	595,0375	2,9751875	48000
103	Prober	142806	595,025	2,975125	48000
104	Sneaky Tit For Tat	142806	595,025	2,975125	48000
105	Negation	142803	595,0125	2,9750625	48000
106	π	142803	595,0125	2,9750625	48000
107	Predator	142803	595,0125	2,9750625	48000
108	Prober 3	142803	595,0125	2,9750625	48000
109	e	142803	595,0125	2,9750625	48000
110	Meta Winner	142803	595,0125	2,9750625	48000
111	Meta Winner Deterministic	142803	595,0125	2,9750625	48000
112	Meta Winner Ensemble	142803	595,0125	2,9750625	48000
113	Meta Winner Memory One	142803	595,0125	2,9750625	48000
114	Meta Winner Finite Memory	142803	595,0125	2,9750625	48000
115	Meta Winner Long Memory	142803	595,0125	2,9750625	48000
116	Meta Winner Stochastic	142803	595,0125	2,9750625	48000
117	Mind Controller	142800	595	2,975	48000
118	Mind Reader	142800	595	2,975	48000
119	Mind Warper	142800	595	2,975	48000
120	Protected Mind Reader	142800	595	2,975	48000
121	Raider	142800	595	2,975	48000
122	Tricky Defector	142800	595	2,975	48000
123	UsuallyDefects	142800	595	2,975	48000
124	Meta Minority	142800	595	2,975	48000
125	Knowledgeable Worse and Worse	141519	589,6625	2,9483125	48000
126	Fortress3	140931	587,2125	2,9360625	48000
127	Fortress4	139436	580,9833333	2,904916667	48000
128	Hopeless	137752	573,9666667	2,869833333	48000
129	Gladstein	133065	554,4375	2,7721875	48000
130	EasyGo	132249	551,0375	2,7551875	48000
131	Harrington	131777	549,0708333	2,745354167	48000
132	Evolved ANN 5 Noise 05	129063	537,7625	2,6888125	48000
133	Naive Prober	127399	530,8291667	2,654145833	48000
134	Darwin	125274	521,975	2,609875	48000

135	Joss	124840	520,1666667	2,600833333	48000
136	Evolved FSM 16	124147	517,2791667	2,586395833	48000
137	EvolvedLookerUp2_2_2	124070	516,9583333	2,584791667	48000
138	Michaelos	124011	516,7125	2,5835625	48000
139	MEM2	123942	516,425	2,582125	48000
140	Evolved FSM 16 Noise 05	123901	516,2541667	2,581270833	48000
141	Mikkelson	123816	515,9	2,5795	48000
142	Limited Retaliate 3	123597	514,9875	2,5749375	48000
143	Limited Retaliate	123573	514,8875	2,5744375	48000
144	Leyvraz	123532	514,7166667	2,573583333	48000
145	Limited Retaliate 2	123479	514,4958333	2,572479167	48000
146	Kluepfel	123470	514,4583333	2,572291667	48000
147	Evolved HMM 5	123325	513,8541667	2,569270833	48000
148	Inverse Punisher	123262	513,5916667	2,567958333	48000
149	Gradual	123042	512,675	2,563375	48000
150	Hard Tit For Tat	123001	512,5041667	2,562520833	48000
151	Inverse	122938	512,2416667	2,561208333	48000
152	Evolved ANN	122688	511,2	2,556	48000
153	Math Constant Hunter	122537	510,5708333	2,552854167	48000
154	GTFT	122489	510,3708333	2,551854167	48000
155	Forgetful Fool Me Once	122468	510,2833333	2,551416667	48000
156	Evolved ANN 5	122427	510,1125	2,5505625	48000
157	DBS	122423	510,0958333	2,550479167	48000
158	Hard Tit For 2 Tats	122416	510,0666667	2,550333333	48000
159	DoubleCrosser	122355	509,8125	2,5490625	48000
160	Level Punisher	122221	509,2541667	2,546270833	48000
161	Evolved FSM 4	121985	508,2708333	2,541354167	48000
162	Grofman	121968	508,2	2,541	48000
163	Hesitant QLearner	121945	508,1041667	2,540520833	48000
164	Soft Go By Majority: 10	121838	507,6583333	2,538291667	48000
165	Grumpy	121755	507,3125	2,5365625	48000
166	Fool Me Once	121545	506,4375	2,5321875	48000
167	Firm But Fair	121323	505,5125	2,5275625	48000
168	Eatherley	121296	505,4	2,527	48000
169	GraaskampKatzen	121284	505,35	2,52675	48000
170	GrudgerAlternator	121272	505,3	2,5265	48000
171	Getzler	121118	504,6583333	2,523291667	48000
172	Soft Go By Majority: 20	121031	504,2958333	2,521479167	48000
173	BackStabber	120883	503,6791667	2,518395833	48000
174	Soft Go By Majority: 5	120795	503,3125	2,5165625	48000
175	General Soft Grudger	120749	503,1208333	2,515604167	48000
176	Soft Go By Majority: 40	120713	502,9708333	2,514854167	48000
177	EvolvedLookerUp1_1_1	120690	502,875	2,514375	48000
178	EugineNier	120518	502,1583333	2,510791667	48000
179	Forgiving Tit For Tat	120477	501,9875	2,5099375	48000
180	Grudger	120467	501,9458333	2,509729167	48000
181	Cave	120232	500,9666667	2,504833333	48000
182	Borufsen	120112	500,4666667	2,502333333	48000
183	Geller Cooperator	119905	499,6041667	2,498020833	48000
184	Soft Go By Majority	119843	499,3458333	2,496729167	48000

185	Forgetful Grudger	119818	499,2416667	2,496208333	48000
186	Forgiver	119642	498,5083333	2,492541667	48000
187	AON2	119577	498,2375	2,4911875	48000
188	Doubler	119546	498,1083333	2,490541667	48000
189	Desperate	119528	498,0333333	2,490166667	48000
190	Delayed AON1	119441	497,6708333	2,488354167	48000
191	Geller	119309	497,1208333	2,485604167	48000
192	Eventual Cycle Hunter	119160	496,5	2,4825	48000
193	Defector	118988	495,7833333	2,478916667	48000
194	Feld	118675	494,4791667	2,472395833	48000
195	Hard Go By Majority: 5	118604	494,1833333	2,470916667	48000
196	Geller Defector	118593	494,1375	2,4706875	48000
197	Adaptive Pavlov 2006	118500	493,75	2,46875	48000
198	Adaptive Pavlov 2011	118481	493,6708333	2,468354167	48000
199	Davis	118470	493,625	2,468125	48000
200	Champion	118412	493,3833333	2,466916667	48000
201	Contrite Tit For Tat	118332	493,05	2,46525	48000
202	Cooperator Hunter	118167	492,3625	2,4618125	48000
203	Black	117815	490,8958333	2,454479167	48000
204	Hard Go By Majority: 40	117653	490,2208333	2,451104167	48000
205	Hard Go By Majority: 20	117562	489,8416667	2,449208333	48000
206	Hard Go By Majority: 10	117535	489,7291667	2,448645833	48000
207	Defector Hunter	117480	489,5	2,4475	48000
208	Alexei	117188	488,2833333	2,441416667	48000
209	DoubleResurrection	116279	484,4958333	2,422479167	48000
210	Appeaser	116136	483,9	2,4195	48000
211	Cycle Hunter	116065	483,6041667	2,418020833	48000
212	CollectiveStrategy	116054	483,5583333	2,417791667	48000
213	Hard Go By Majority	115976	483,2333333	2,416166667	48000
214	Adaptive Tit For Tat	115865	482,7708333	2,413854167	48000
215	Gradual Killer	115613	481,7208333	2,408604167	48000
216	Cooperator	113628	473,45	2,36725	48000
217	Colbert	113128	471,3666667	2,356833333	48000
218	Calculator	112442	468,5083333	2,342541667	48000
219	AdaptorBrief	112105	467,1041667	2,335520833	48000
220	AdaptorLong	111863	466,0958333	2,330479167	48000
221	Cycler DC	111826	465,9416667	2,329708333	48000
222	Alternator Hunter	110968	462,3666667	2,311833333	48000
223	Cycler CCCDCD	110706	461,275	2,306375	48000
224	Cycler DDC	110384	459,9333333	2,299666667	48000
225	Bully	109721	457,1708333	2,285854167	48000
226	Cycler CCD	109332	455,55	2,27775	48000
227	Cautious QLearner	109200	455	2,275	48000
228	Arrogant QLearner	108176	450,7333333	2,253666667	48000
229	Average Copier	108172	450,7166667	2,253583333	48000
230	Cycler CCCD	107976	449,9	2,2495	48000
231	Better and Better	107814	449,225	2,246125	48000
232	Cycler CCCCCD	107563	448,1791667	2,240895833	48000
233	ALLCorALLD	107293	447,0541667	2,235270833	48000
234	Adaptive	105948	441,45	2,20725	48000

235	Bush Mosteller	105598	439,9916667	2,199958333	48000
236	Anti Tit For Tat	100634	419,3083333	2,096541667	48000
237	Aggravater	100500	418,75	2,09375	48000
238	Alternator	100375	418,2291667	2,091145833	48000
239	AntiCycler	98116	408,8166667	2,044083333	48000

FONTE: Autor (2019)