Implementação de Comunicação entre Agentes Autônomos para Aprendizagem de Comportamentos Colaborativos

Renato A. Nobre, Guilherme N. Ramos, Departamento de Ciência da Computação, Universidade de Brasília

***Resumo*—Sistemas de múltiplos robôs são capazes de completar muitas tarefas de maneira mais rápida e confiável que sistemas de apenas um robô. A comunicação entre os mesmos pode conseguir multiplicar suas capacidades e efetividades. Este relatório discute o uso da comunicação na redução de efeitos indesejados em sistemas multiagentes estocásticos, com múltiplos robôs aprendendo em paralelo enquanto interagem entre si. Dois grandes problemas, ruído e localidade, são endereçados utilizando métodos de comunicação direta em uma abordagem "quadro-negro". A metodologia implementada é testada em um simulador de seleção de comportamentos em múltiplos agentes autônomos com aprendizagem por reforço, o qual simula um ambiente clássico de caça-predador. A comunicação é utilizada para compartilhar, entre os predadores, o mapa de probabilidades da posição da presa, para tratar o problema do ruído; e valores de aprendizagem por reforço, para diminuir o problema de localidade. Os resultados obtidos mostraram que o erro proveniente do ruído diminui com a comunicação dos mapas de probabilidades. Já comunicação do aprendizado pode ser capaz de aumentar as politicas agressivas e diminuir as de fuga, demonstrando também impacto nas seleções de comportamentos nos experimentos, onde os agentes preferem utilizar políticas iguais ao invés de politicas complementares. No entanto, as politicas aprendidas pelos agentes com comunicação não geraram comportamentos cooperativos em todos os experimentos realizados.**

Keywords*—*aprendizado compartilhado, mapas de probabilidades, comunicação, caça-predador, sistema multiagentes, inteligência artificial, programação Bayesiana, Q-learning, aprendizagem por reforço.

I. Introdução

Incertezas são inerentes a aplicações no mundo-real, onde as informações não podem ser medidas diretamente e as medições são frequentemente imprecisas [2]. Por exemplo, a robótica, amplamente estudada no campo de inteligência artificial distribuída [18], é inerentemente estocástica, em outras palavras, é aleatória e há indeterminações presentes. Os seus modelos matemáticos descrevem a realidade de forma simplificada, os sensores são, na maioria das vezes, imprecisos e imprevisíveis, e as formas de representação dos estados (do conhecimento acerca do ambiente) são limitadas [12].

Apesar de não haver consenso na literatura, é necessário utilizar uma definição de agente para limitar o escopo do trabalho. Desta forma, consideraremos um agente como uma entidade que percebe o ambiente por meio de sensores e atua neste ambiente, de forma autônoma, por meio de atuadores [7]; e agentes racionais como agentes que buscam ações que maximizam uma determinada medida de desempenho, que é a função do estado do ambiente [13]. Quando múltiplos agentes coexistem e interagem em um ambiente dinâmico de maneira cooperativa ou competitiva, ele é classificado como multiagente [14].

Agentes cooperativos combinam seus esforços para atingir um objetivo em comum, permitindo que os objetivos de múltiplos agentes sejam cumpridos simultaneamente [13]. Um cenário comumente aplicado em sistemas multiagentes é o de caça-predador, onde a caça deve evitar ser capturada, enquanto os predadores visam capturá-la [1]. Um exemplo deste tipo de cenário é o caso de múltiplos drones autônomos caçando um fugitivo, onde o fugitivo é uma presa em movimento, sua localização é conhecida, e quando capturada, implicará em uma atribuição de recompensa a todos os drones. Apesar de não possuir um único formato o problema de caça-predador tradicionalmente possui um ambiente discreto, uma caça e quatro predadores [3]. Como o objetivos dos predadores é o mesmo, esse cenário é caracterizado como sistema multiagente cooperativo [13]. Nestes ambientes, um comportamento considerado bom em determinado momento pode deixar de sê-lo em outro, de modo que um agente racional precisa de mecanismos para se adaptar. Um destes mecanismos é o aprendizado, neste trabalho entendido como a capacidade do agente de melhorar seu desempenho utilizando experiência adquirida ao longo do tempo [7].

O aprendizado por reforço é uma das principais abordagens, no qual o agente descobre quais ações devem ser executadas de forma a maximizar as recompensas numéricas recebidas do ambiente [13]. Neste cenário os agentes não possuem informações das melhores ações a serem realizadas, descobrindo por conta própria ao interagir com o ambiente [10].

Na robótica, a utilização de simuladores é frequente, uma vez que simulações computacionais costumam ser mais rápidas e seguras que testes físicos [21]. Jogos digitais, providenciam um ambiente de simulações ideal para a experimentação e estudo da inteligência artificial [16]. Um jogo clássico que cabe ao estudo do problema caça-predador em sistemas multiagentes é o *Pac-Man*, onde o jogador, representado pela personagem *Pac-Man*, tem que comer todas as comidas disponíveis no meio, e evitar ser capturado por fantasmas [1].

Este trabalho utiliza como base uma plataforma para seleção de comportamentos em sistemas com as características descritas, porém os agentes racionais não agem em grupo para capturar a caça em todos os experimentos realizados na plataforma. Portanto, é interessante analisar se a aplicação de métodos, como comunicação entre agentes, no simulador são

capazes de gerar políticas cooperativas em todas as simulações. É possível que, ao inserir métodos de comunicação entre os agentes, eles explorem mais frequentemente a cooperação e alcancem resultados melhores.

Visa-se principalmente na implementação de dois métodos de comunicação. O primeiro método foca em diminuir o erro de precisão dos sensores, o ruído do sistema. O problema do ruído aumenta as incertezas do sistema, e é proveniente da falta de sensores confiáveis, sendo um dos maiores limitadores de aprendizado em sistemas robóticos [15]. O segundo método visa resolver o problema de localidade do sistema distribuído, ou seja, o fato dos agentes decidirem realizar o objetivo por si só, sendo um fator de restrição à cooperação a nível de grupo entre os agentes [8].

O relatório está organizado do seguinte modo. A Seção II desenvolve conceitos teóricos importantes para o entendimento da plataforma e do trabalho; a Seção III introduz o conceito de sistemas multiagentes e a plataforma utilizada; a Seção IV desenvolve os conceitos e métodos de comunicação implementados neste trabalho; a Seção V introduz as hipóteses do roteiro experimental e seus resultados; por ultimo, a Seção VI finaliza o relatório, sumariando os resultados obtidos e sugerindo trabalhos futuros.

II. Aprendizado por Reforço e Programação Bayesiana

Em ambientes estocásticos, meios aleatórios onde incertezas estão presentes, o agente não tem como se basear em exemplos prévios, e começa a simulação sem qualquer informação do lugar inserido e nenhuma função de recompensa [10]. No entanto, se o agente tentar movimentos aleatórios, ele consegue, eventualmente, construir um modelo preditivo do meio, recebendo uma recompensa r e aprendendo de experiências prévias em que as ações possam melhorar a recompensa futura [7].

Na inteligência artificial, aprendizado por reforço representa uma categoria de técnicas computacionais para que os agentes melhorem seu desempenho ao longo do tempo a partir das suas iterações com o ambiente no qual está inserido [10]. Sendo assim a tarefa do aprendizado por reforço é usar tais recompensas r para aprender uma função ótima de comportamento, ou seja, uma função que consegue atingir o objetivo com maior recompensa no menor período de tempo. O agente recebe uma função de ação-valor, com base na recompensa esperada de realizar uma ação em um determinado estado do ambiente. Isto é denominado *Q-learning* [7].

A. Q-learning

O algoritmo do Q-learning foi desenvolvido como algoritmo de controle de agentes com a simplicidade e a possibilidade de transicionar uma função V que estima o valor de um estado atual s_t para uma política de controle otimizada, que estima valores de pares estado-ação $Q(s_t, a_t)$ [10]. Assim, um agente não armazena a informação de quanta recompensa pode ser recebida a partir de um estado, mas sim que poderá ser recebida se, em um determinado estado s_t, o agente executar a ação a_t [1].

O algoritmo de Q-learning utiliza o fator de desconto γ, limitado ao intervalo $0 \leq \gamma \leq 1$, que permite fornecer maior relevância para recompensas imediatas ou futuras [2]. Quando γ se aproxima de 0 o agente busca maximizar recompensas a curto prazo, analogamente ao se aproximar de 1 recompensas a longo prazo serão valorizadas [1]. Outro fator utilizado pelo Q-learning é a taxa de aprendizagem α contida em $0 < \alpha < 1$ que, por sua vez, tem o papel de limitar a aprendizagem por iteração, ou seja, a variação máxima do valor $Q(s_{\text{t-1}}, a_{\text{t-1}})$ [2].

A Equação (1) apresenta o Q-learning com um passo.

$$Q(s_{\text{t-1}}, a_{\text{t-1}}) = Q(s_{\text{t-1}}, a_{\text{t-1}}) + \alpha\varphi \tag{1}$$

Onde a φ é definida como:

$$\varphi = [r_{\text{t}} + \gamma \max{}_{\text{a}} Q(s_{\text{t}}, a) - Q(s_{\text{t-1}}, a_{\text{t-1}})] \tag{2}$$

Considere, por exemplo, uma situação de caça-predador como a do *Pac-Man*. Se o *Pac-Man* está em um estado s a uma célula de distância de um fantasma, e realiza uma ação a' que o faz ser capturado, o valor de $Q(s, a')$ é atualizado negativamente. Portanto, na próxima vez que o *Pac-Man* estiver no estado s, considerando que existam ações melhores, o Q-learning escolherá a ação que possui o maior valor Q, evitando assim a ação a'.

No entanto, as Equações (1) e (2) são computacionalmente inviáveis em ambientes contínuos devido a infinidade de pares estado-ação [2]. Neste caso uma alternativa é aplicar métodos de aproximação de funções para estimar $Q(s_t, a_t)$.

B. Q-learning com Aproximação de Funções

Em sistemas estocásticos, pode se tornar inviável a execução do algoritmo representado pela Equação (1) devido a limitações de hardware e tempo, como tamanho de memória e poder de processamento [1]. Este problema pode ser corrigido para ambos os casos, usando aproximação de funções para estimar o valor de $Q(s_t, a_t)$. Deste modo, ao invés de guardarmos os valores de todos possíveis pares de estado-ação, armazena-se somente um vetor de parâmetros θ, usado para calcular a estimativa do valor $Q(s_t, a_t)$ [1]. O aprendizado é então realizado ao atualizar as componentes θ_i do vetor, utilizando as recompensas recebidas pelo agente.

Entre os métodos de aproximação de função, uma escolha comum é o método de aproximação linear [10], onde se define um vetor de características $\phi(s)$, do mesmo tamanho de θ, a partir do estado atual do ambiente. Assim, o parâmetro θ_i define o peso que cada característica $\phi(s)$, um valor no intervalo $0 \leq \phi_i(s) \leq 1$, terá no cálculo da aproximação de $Q(s, a)$ [1].

Por exemplo, no cenário do *Pac-Man*, o vetor de características pode ser composto por duas características distintas, a primeira assumindo o valor 0 ou 1, indicando se o fantasma encontra-se a uma célula de distancia do *Pac-Man* e a segunda representa a probabilidade do fantasma não capturar o *Pac-Man*. Um vetor de parâmetros θ com valor negativo no primeiro parâmetro $\phi_1(s)$, e positivo no segundo, $\phi_2(s)$, indica que $\phi_1(s)$ afeta negativamente o valor $Q(s_t, a_t)$ do par estado ação, e $\phi_2(s)$ afeta positivamente.

Logo, a partir dos vetores θ e ϕ, o valor estimado do par estado-ação é calculado. Na aproximação linear o aprendizado ocorre atualizando o vetor θ de acordo com a recompensa recebida [2].

C. Programação Bayesiana

Algoritmos de aprendizado por reforço selecionam, para o estado atual do sistema, o comportamento mais adequado de acordo com seu objetivo. No entanto, tais algoritmos necessitam que o estado seja uma variável conhecida, o que não acontece em ambientes estocásticos [2]. Processos de tomadas de decisão que consideram incertezas do ambiente têm sido estudados cada vez mais por pesquisadores de inteligência artificial [12].

Programação Bayesiana é um *framework* para desenvolver sistemas inteligentes, auxiliando nos processos de tomada de decisões que levam em consideração as incertezas de um ambiente estocástico. O formalismo matemático é genérico suficiente para reinterpretar modelos probabilísticos clássicos [19]. Seu elemento fundamental é a proposição lógica, uma hipótese h que pode ser verdadeira ou falsa, sendo todas as proposições dependentes do conhecimento prévio do sistema, representado por π. Assim proposições lógicas são sempre condicionadas em π, ou seja, $P(h|\pi)$ [20]. Outro conceito fundamental na Programação Bayesiana é o de variável discreta: um conjunto X de proposições exaustivas e mutualmente exclusivas.

Qualquer programa Bayesiano contém duas partes: uma *descrição*, conjunto de distribuições probabilísticas de todas as variáveis pertinentes, usando conhecimento anterior π e dados experimentais δ; e uma *questão* que por sua vez, é uma distribuição probabilística calculada usando a *descrição* [20].

Portanto, é dividida a *descrição* em três conjuntos de variáveis: S, representando as variáveis em que as probabilidades são calculadas; K, como as variáveis que podem ser observadas; e U como as variáveis não conhecidas [19]. Por exemplo, considere a *descrição* os conjuntos de probabilidades de, "*Pac-Man* estar próximo do fantasma", "*Pac-Man* próximo a pílula" e "fantasma próximo a pílula". A *questão* é definida então pela probabilidade do *Pac-Man* estar próximo do fantasma dado a probabilidade do *Pac-Man* e do fantasma estarem próximas da pílula. Em seguida, essa estimação pode ser fornecida em outras partes do sistema inteligente, tal como na seleção de ações [2].

III. Plataforma de Simulação do Aprendizagem por Reforço para Sistemas Multiagentes

A implementação de comunicação entre agentes autônomos tem como base o sistema desenvolvido no trabalho de *Seleção de comportamentos em múltiplos agentes autônomos com aprendizagem por reforço em ambientes estocásticos* [1].

O trabalho propõe um algoritmo para sistemas multiagentes estocásticos, utilizando programação Bayesiana para estimação de estados, e Q-learning com aproximação de funções para prover aos agentes a capacidade de aprender e selecionar comportamentos mais adequados. Para propósito do trabalho citado, foi definido comportamento como o conjunto de ações realizados por um agente de acordo com o estado do ambiente. Tal definição será usada também para este trabalho.

A. Sistemas Multiagentes

O subcampo da inteligência artificial que provê princípios e mecanismos para construção de sistemas complexos e coordenação de comportamento de agentes é o de Sistemas Multiagentes [3]. Embora não tenha um conceito geral definido para um agente em inteligência artificial [7], para o propósito desta pesquisa, consideraremos um agente como uma entidade inserida em certo meio, com objetivos e ações bem definidos [4].

B. O Problema de Localidade e do Ruído

Um desafio chave para sistemas multiagentes distribuídos, é atingir cooperação a nível de grupo entre os agentes [8]. Sistemas completamente distribuídos em que cada agente aprende independentemente introduzem dificuldades como ambientes não estacionários, em que a medida que os agentes aprendem, seus comportamentos mudam, resultando em inconsistências; e problemas de atribuição de crédito, que causa agentes realizarem objetivos sozinhos. Neste problema, como cada agente recebe suas recompensas individualmente, se um dos agentes aprende a realizar o objetivo sozinho e consegue resultados satisfatórios sem a necessidade dos outros do grupo, ele poderá tender a sempre realizar os objetivos e os outros fantasmas não colaborarão [14].

Tal comportamento é denominado *localidade* do sistema distribuído, usualmente este comportamento não é desejado, tendo em vista que em problemas de múltiplos agentes com o mesmo objetivo, resultados melhores podem ser obtidos com a cooperação [8].

Outro problema bastante frequente em ambientes estocásticos e em sistemas robóticos é a questão da presença do ruído em que atrapalha as medidas dos sensores em inúmeras maneiras, e consequentemente limita a informação que pode ser extraída pelas medições [12]. A falta de sensores precisos e confiáveis é indiscutivelmente o maior problema de pesquisadores de controle de agentes e aprendizado, sendo um dos maiores maiores fatores de limitação para a aprendizagem em tempo real [15].

C. Simulador de Pac-Man

Foi utilizado o simulador do jogo *Pac-Man*, desenvolvido para disciplinas de inteligência artificial pelos professores John DeNero e Dan Klein da Universidade de Califórnia, Berkeley, denominado *The Pac-Man Projects* [1].

Uma distinção importante do simulador é a existência de um sistema de pontuação que pode ser utilizado para atribuir as recompensas recebidas pelos agentes e, assim, usada no aprendizado por reforço [1]. Tal sistema de pontuação presente no simulador é único, isto significa que, quanto menor a pontuação do sistema, melhor foi o desempenho dos fantasmas, e quanto maior a pontuação, melhor o desempenho do *Pac-Man*.

[1] Disponível em: http://ai.berkeley.edu/project_overview.html

D. Especificações do Sistema

A estrutura do sistema foi desenvolvido para evitar acoplamento entre o algoritmo de aprendizado e o simulador, utilizado dois módulos, o controlador e o adaptador [1]. Assim, é possível realizar teste em sistemas robóticos ou em outros simuladores com pouca alteração de código.

O controlador recebe informações de medições e recompensas fornecidas pelo adaptador, repassa tais dados para o módulo de aprendizado, executa os comportamentos e envia as ações de volta. Já o sistema adaptador é responsável por conciliar o ambiente para respeitar as interfaces disponibilizadas pelo controlador [1].

O sistema de comunicação utilizado para transferir mensagens entre o módulo de comunicação e adaptador foi implementado usando a biblioteca *ZeroMQ3*[2]. Sua escolha foi realizada devido à facilidade de implementar diversas arquiteturas de comunicação com poucas modificações, bem como a abstração do protocolo de envio e recebimento de dados *TCP/IP*, permitindo que os módulos possam ser executados de maneiras simples em máquinas diferentes [1].

Por fim, o sistema utilizado apresenta problemas típicos de sistemas multiagentes distribuídos, como os descritos na Seção III-B. Para lidar com estes comportamentos, métodos de comunicação foram implementados.

IV. Proposta e Desenvolvimento

Comunicação entre agentes é capaz de multiplicar suas capacidades e eficácias [4]. Qualquer comportamento observado, bem como suas consequências, pode ser interpretado como uma forma de comunicação.

O sistema desenvolvido em [1], quando usado com fantasmas racionais, para propósito desta pesquisa definidos como fantasmas que passaram por um processo de aprendizagem, e um *Pac-Man* que não realizou tal processo, pode ser classificado como um *sistema multiagente homogêneo sem comunicação* [3]. Em tal sistema todos os agentes possuem a mesma estrutura interna, incluindo o objetivo. No entanto, ao aplicarmos comunicação podemos transformá-lo em um *sistema multiagente homogêneo com comunicação*, e assim agentes, possivelmente, conseguirão coordenar seus objetivos de forma mais eficaz do que anteriormente.

Para desenvolver o conceito da implementação da comunicação, consideremos suas formas e definições. A *comunicação indireta* é baseada na observação do efeito do comportamento de outros agentes no ambiente, em outras palavras, é a comunicação perceptível da modificação de um ambiente [8]. Em contraste, temos a *comunicação direta*, um ato de comunicação puro, com o propósito de transmitir informação para um agente destinatário. Para o propósito desta pesquisa, implementou-se a comunicação direta com o objetivo de reduzir os problemas de ruído e localidade. A escolha da comunicação direta foi favorecida devido a os métodos de comunicação indireta não serem eficientes em situações em que a comunicação não gera alterações no meio, pois para implementá-los os agentes necessitariam, de certo modo, alterar o ambiente [4].

[2]Disponível em: http://zeromq.org

A transmissão de mensagens entre os agentes foi implementada em uma abordagem "quadro-negro", onde todos os agentes enviam suas informações para um módulo que processará os dados e redistribuirá de volta para os agentes [13]. A abordagem foi favorecida em relação a abordagem de transmissão "ponto a ponto" de mensagens, de um agente para outro agente específico [3]. O favorecimento se deve ao sistema já possuir um módulo controlador centralizado que funciona idealmente para a abordagem.

Para realizar a transmissão de mensagens entre o módulo controlador e os agentes foram definidos dois tipos de protocolos de mensagem no módulo de comunicação: *Solicitação* e *Atribuição*. Os protocolos do tipo de *Solicitação*, pedem para o agente a informação que necessita e responde quando a informação for recebida. Por outro lado os protocolos de *Atribuição* realizam o devido tratamento dos dados provenientes dos métodos de *Solicitação* e distribui o dado tratado aos agentes.

Dois tipos de comunicação direta foram implementados: comunicação da posição estimada do *Pac-Man*, e comunicação dos parâmetros de estados, comportamento e recompensa dos fantasmas.

A. Implementação da Comunicação do Mapa de Probabilidades

Em sistemas multiagentes, a habilidade de detectar e distinguir corretamente agentes de um grupo ou de outro, obstáculos e outras diversas características do ambiente é crucial para a maioria das tarefas e objetivos [8].

Na plataforma utilizada, a habilidade dos agentes detectarem suas posições, e detectarem uns aos outros no ambiente é realizada com mapas de probabilidade. Um mapa de probabilidade é um modelo probabilístico do ambiente inserido em que cada posição de uma matriz $M_{i,j}$ possuirá um valor x contido em $0 < x < 1$, e o somatório de todas as suas posições é sempre igual a 1, representado pela Equação (3). Portanto, o maior valor x das posições $m_{k,l}$ é onde existe a maior probabilidade de um agente estar localizado.

$$\sum_{k=1}^{i}\sum_{l=1}^{j} m_{k,l} = 1 \tag{3}$$

Cada agente da simulação possui mapas de probabilidade de todas as entidades da simulação. Um agente possui um mapa de probabilidade exclusivo para a sua posição, e para a posição dos outros $n-1$ agentes, onde n é o número de agentes inseridos no meio. Exemplificando, suponha que uma simulação é executada com três fantasmas e um *Pac-Man*, cada fantasma possuirá um mapa de probabilidades de sua posição, um para cada um de seus aliados e um para a posição do *Pac-Man*, ou seja, cada fantasma possuirá quatro mapas de probabilidades distintos, um para cada entidade.

A presença de ruído proveniente de sensores é um dos maiores desafios do aprendizado em sistemas, podendo assim dificultar a habilidade de detecção dos agentes. Para simular esse problema proveniente dos sensores, o simulador provém de um sistema de implementação de ruído nos mapas de

probabilidade, podendo aumentar o erro de estimativa dos agentes.

Portanto, a comunicação do mapa de probabilidades foi implementada nos fantasmas com o objetivo de diminuir o erro de estimação proveniente do ruído na detecção do *Pac-Man*. Ou seja, cada fantasma comunica apenas o mapa da probabilidade da posição do *Pac-Man*. Esta comunicação ocorre de maneira em que o módulo controlador chama o protocolo de *Solicitação*, requerindo para cada um dos agentes que comunique a sua estimativa da posição do *Pac-Man*. Quando todos os fantasmas tiverem comunicado a própria estimativa, o controlador realizará a união dos mapas.

A união dos mapas é feita que para cada posição $m_{i,j}$ do mapa de um agente seja multiplicado pela mesma posição $m_{i,j}$ do mapa dos outros agentes. Assim um novo mapa M' será gerado pela multiplicação de cada posição $m_{i,j}$ de cada mapa dos agentes. No entanto, ao multiplicarmos probabilidades necessitamos normalizá-las, isto é, precisamos dividir cada valor do mapa pelo somatório de todos os valores, para que o somatório de todas as posições $m_{i,j}$ seja sempre 1.

Em ambientes computacionais a multiplicação de duas probabilidades muito pequenas pode ser arredondada para zero devido ao erro de aproximação. Para evitar esse comportamento indesejado, a multiplicação foi substituída por um método equivalente, usando a soma de logaritmos naturais na base e. No fim dos cálculos o mapa M' é distribuído para os agentes pelo protocolo de *Atribuição*.

Para testar se a abordagem da união dos mapas de probabilidades era realmente capaz de diminuir o erro gerado pelo ruído, foram implementados métodos para cálculo do erro quadrático médio das simulações.

Os cálculos dos erros ξ de cada simulação quando há comunicação do mapa de probabilidades foi realizado da seguinte maneira: a cada instância da simulação era obtida a diferença absoluta da maior probabilidade do mapa conjunto com a posição verdadeira do *Pac-Man* ao quadrado. Portanto, no final do jogo era obtido o somatório dos erros de cada instância t dividido pelo número de instâncias, descrito na Equação (4). Esse procedimento se repete por k jogos e no fim da simulação o erro é equivalente ao o somatório do erro dos jogos ξ_j divido pela quantidade k de jogos, assim representado pela Equação (5).

$$\xi_j = \frac{\sum_{t=1}^{n} \xi_t}{n} \tag{4}$$

$$\xi = \frac{\sum_{j=1}^{k} \xi_j}{k} \tag{5}$$

Similarmente, em casos que não há a presença da comunicação, o erro do jogo é calculado sendo o somatório dos maiores erros entre os agentes. O erro de cada agente é calculado pela diferença de estimativa de maneira similar ao método com comunicação. No fim da simulação, o erro equivale ao somatório do erro dos jogos divido pela quantidade de jogos.

B. Implementação da Comunicação do Aprendizado

A comunicação do aprendizado tem como objetivo diminuir o problema de localidade e resolver a questão da atribuição de créditos, para que os agentes realizem ações como um grupo com base em políticas cooperativas.

Consideraremos políticas cooperativas como políticas complementares para atingir o objetivo. O *framework* de *Pac-Man*, define três tipos de comportamento em que o fantasma pode executar, *Perseguição*, em que o fantasma se move de forma a alcançar a posição futura do *Pac-Man*; *Busca*, de caráter predatório, o agente se move em linha reta em relação a caça; e por último a *Fuga* onde o agente procura se afastar do predador. Portanto, definiremos políticas cooperativas, políticas em que os agentes escolhem prioritariamente comportamentos complementares, ou seja *Perseguição* e *Busca*.

Para adereçar o problema de atribuição de créditos e localidade, foram desenvolvidos protocolos de comunicação que transferem as informações do aprendizado. O módulo controlador, ao executar o protocolo de *Solicitação*, requere de cada agente fantasma, que comunique seu estado s_t, o estado prévio s_{t-1}, o comportamento executado c, e a recompensa r recebida pelo comportamento. Ao controlador receber as informações de aprendizado de todos os f fantasmas, ele realiza a atualização do aprendizado chamando o protocolo de *Atribuição*, sendo assim, cada fantasma aprende n vezes por iteração.

A atualização do aprendizado ocorre quando o protocolo de *Atribuição* recebe as variáveis armazenadas pelo protocolo de *Solicitação* e chama um método do módulo de aprendizagem. Este método recebe como parâmetro tais variáveis e atualiza o peso φ, representado pela Equação (2). O método é chamado $f - 1$ vezes por cada fantasma, cada vez atualizando o aprendizado com os parâmetros de um outro agente aliado.

V. Resultados Experimentais

Com base nos problemas citados na Seção III-B, e com os métodos implementados no sistema descritos na Seção IV foram elaboradas as seguintes perguntas:

1) A comunicação dos mapas de probabilidades é capaz de diminuir o erro proveniente do ruído?
2) A aplicação da comunicação dos mapas de probabilidades pode ser capaz de diminuir o número de instancia das simulações no caso da presença de ruído?
3) Será possível com o uso de recompensa compartilhada diminuir a localidade do sistema?
4) É possível com a comunicação do aprendizado que a pontuação final dos jogos, a favor dos fantasmas, aumente significativamente quando o agente do *Pac-Man* é eficiente?
5) Com os dois tipos de comunicação, mapa de probabilidades e comunicação do aprendizado, é possível que os agentes gerem políticas cooperativas em todas as execuções, quando colocados contra um *Pac-Man* aleatório? E contra um *Pac-Man* eficiente?

Com o propósito de responder as perguntas apresentadas foram elaborados experimentos. Consideraremos um experimento como o conjunto de 30 simulações realizadas com os mesmos parâmetros de simulação. O valor de 30 simulações foi definido pela Equação (6), que estima um tamanho amostral dado que não temos conhecimento da variância, nem da

proporção populacional, com $\delta = 0.95$ e $\epsilon \approx 0.179$. Onde δ é o valor do intervalo de confiança, z_δ é o valor da distribuição normal padrão para um intervalo de confiança δ, e ϵ é o erro amostral máximo [11].

$$n \approx \frac{z_\delta^2}{4\epsilon^2} \tag{6}$$

Independentemente do experimento realizado, todas as simulações foram feitas com 100 jogos de aprendizado, 15 jogos de teste, e quando há presença de um ruído n, está contido entre o intervalo $-3 < n < 3$, assim como definido em [1]. As simulações foram feitas com um ambiente de simulação reduzido, comparado com o mapa clássico do *Pac-Man*. A escolha do mapa reduzido se deve a redução efetiva de tempo para a execução dos experimentos.

Os agentes de *Pac-Man* utilizados para os experimentos foram desenvolvidos e descritos em [17]. O *Pac-Man* aleatório utilizado, denominado no simulador como *RandomPacmanAgentTwo*, é um agente pseudo-aleatório que escolhe uma ação e a realiza até a ação não estar mais disponível, ao encontrar uma parede bloqueando-o, ou possuir mais de três ações possíveis, que no caso, haverá uma probabilidade para selecionar uma destas ações. A utilização do *Pac-Man* aleatório foi evitada nas simulações descritas, devido ao fato do agente se manter frequentemente em uma mesma região, ou posição do mapa. Outro *Pac-Man* utilizado, é o *BFSPacmanAgent*, que realiza um procedimento de busca por largura, nos alimentos do mapa. E por ultimo, o *Pac-Man* utilizado considerado como eficiente, é denominado *NimblePacmanAgent*, que se alimenta quando está seguro e foge quando está em perigo, analisando sua posição em relação a dos fantasmas. Sua eficiência foi analisada em comparação aos outros agentes *Pac-Man* disponíveis no simulador, apresentando resultados satisfatoriamente melhores em relação a sua pontuação obtida no jogo.

A. Comunicação do Mapa de Probabilidades

Para responder a Pergunta 1, foram criados dois experimentos. Ambos os experimentos com quatro agentes fantasmas, por ser a quantidade clássica em um problema de caça-predador [3], e havendo a presença do ruído, sendo a única diferença entre os dois a comunicação, ou não, dos mapas.

No caso dos experimentos que houve a comunicação do mapa de probabilidades o erro quadrático médio foi de $5,772$, e nos casos sem a comunicação este erro aumenta para $7,183$. Diminuindo o erro proveniente do ruído em aproximadamente $19,64\%$.

Respondendo a Pergunta 2, foram realizados experimentos com quatro agentes fantasmas. Dois experimentos foram realizados para o *BFSPacmanAgent*, e os outros dois para o *NimblePacmanAgent*, sendo para ambos, a única diferença entre os dois experimentos a presença, ou não, da comunicação dos mapas.

Mesmo com o erro quadrático médio diminuindo, a quantidade de instancias dos jogos não aparenta mudança efetiva. A média de instancias dos experimentos realizados com o *BFSPacmanAgent*, com presença de comunicação, fica em aproximadamente $37,311$, ainda bem próximo de quando não há a comunicação, que fica em média de $38,318$. No caso do *NimblePacmanAgent*, o número de instancias fica décimos maior quando há comunicação em comparação de quando não há, ficando em $43,578$ e $43,225$ respectivamente.

B. Comunicação do Aprendizado

Com a comunicação da recompensa compartilhada, visa-se resolver o problema da atribuição de crédito e diminuir a localidade do sistema. Para responder a Pergunta 3, foram realizados experimentos com quatro fantasmas e sem presença de ruído, novamente utilizando apenas o *BFSPacmanAgent* e o *NimblePacmanAgent*. Para cada agente *Pac-Man* foram realizados dois experimentos, um havendo comunicação do aprendizado, e o outro não.

Nestes experimentos, para cada simulação, foi analisada a probabilidade de cada agente selecionar certo comportamento. Se a probabilidade do agente de selecionar um comportamento, fosse majoritária, maior que 50%, em relação as outras analisadas, considerou-se o comportamento como o principal do agente na simulação. O valor foi escolhido para evitar incertezas na definição do comportamento do agente, visto que existem três comportamentos o majoritário seria maior que 33%. No entanto, analisa-se quando a probabilidade de um comportamento é necessariamente maior que a soma dos outros dois, pois desta forma, tem-se maior precisão da escolha do comportamento do agente. Portanto, se nenhum dos comportamentos do agente possuísse probabilidade maior que 50%, definiu-se seu comportamento final como indeterminado. Foi realizada então a média dos comportamentos selecionados pelos agentes no experimento. A Tabela I sumariza a relação dos experimentos, e a média de cada comportamento.

Tabela I. RESULTADO DOS EXPERIMENTOS DE COMUNICAÇÃO DO APRENDIZADO

	BFSPacmanAgent		*NimblePacmanAgent*	
Comunicação	Ausente	Presente	Ausente	Presente
Perseguição	1,400	1,233	0,967	1,500
Busca	1,267	1,266	1,267	1,400
Fuga	1,233	1,366	1,433	0,900
Indeterminado	0,100	0,133	0,300	0,200

Uma análise mais detalhada mostra que para os casos dos experimentos com o *NimblePacmanAgent*, a média de um agente selecionar um comportamento de fuga diminui de aproximadamente $1,433$ para $0,900$. Enquanto os comportamentos de busca e perseguição aumentam de $1,267$ para $1,400$ e $0,967$ para $1,500$ respectivamente. Resumindo, o comportamento de fuga diminui em $37,19\%$, o de busca aumenta em $10,50\%$ e o de perseguição em $55,11\%$. Outra análise realizada sob o experimento foi a respeito da quantidade de simulações em que o os agentes escolhem políticas cooperativas. Notou-se que no experimento sem a presença da comunicação os agentes escolheram 11 vezes políticas cooperativas, e 3 vezes políticas iguais, em que todos os agentes escolhem no final da simulação a mesma política, sendo uma das 3 vezes todos com políticas de fuga. Com a comunicação, os agentes escolheram menos vezes políticas complementares, totalizando 9 vezes, mas por outro lado, a quantidade de políticas iguais

aumenta para 14 vezes, sendo três vezes com a política de fuga. O que indica que agentes escolhem mais vezes politicas iguais quando há a presença da comunicação.

Para os experimentos com o *BFSPacmanAgent*, a média de um agente selecionar um comportamento de fuga com a comunicação, aumenta de $1,233$ para $1,367$, a seleção do comportamento de perseguição diminui de $1,400$ para $1,233$ e a de busca se mantém em $1,670$ para ambos os casos. Novamente foi realizada a análise de escolha de políticas, notou-se neste caso, que para quando não há comunicação, os agentes nunca selecionaram todas as políticas iguais, e selecionaram 20 vezes políticas complementares. Por outro lado quando há comunicação o número de políticas complementares diminui um pouco, sendo escolhidos em 16 vezes, e as simulações com políticas iguais ocorrem 11 vezes, sendo 6 com o a política de fuga. O mesmo comportamento foi observado no *NimblePacmanAgent* em que as politica iguais são escolhidas prioritariamente sob as politicas complementares, o que pode ser um possível indicativo de que a seleção de politicas agressivas iguais apresentam resultados melhores.

Respondendo a Pergunta 4 foi analisada a pontuação final dos jogos no caso da presença de comunicação do aprendizado. A pontuação do jogo com o *BFSPacmanAgent*, diminui de $59,346$, quando há comunicação, para $45,358$, quando não há. Isto significa que a presença de comunicação diminuiu a pontuação dos fantasmas. E o mesmo tipo de comportamento ocorre para o *NimblePacmanAgent* que, com a recompensa compartilhada, apresenta uma pontuação de $463,207$, que diminui para $444,806$ no experimento sem comunicação.

C. Comunicação Conjunta

Por último, foi analisada comunicação conjunta com o propósito de responder a Pergunta 5. Foram realizados oito experimentos com a presença de ruído, quatro utilizando dois fantasmas e quatro utilizando três. Esta quantidade de fantasmas reduzida foi definida devido ao fato de apenas duas políticas cooperativas existirem, assim para dois fantasmas, o ideal seria que cada fantasma escolhesse uma política complementar a outra.

Os primeiros quatro experimentos foram realizados com dois fantasmas e os agentes *RandomPacmanAgentTwo* e *NimblePacmanAgent*. Novamente, para efeito de comparação, dois dos experimentos foram realizados sem nenhuma comunicação. A Tabela II sumariza os resultados destes experimentos.

Para os experimentos com o *Pac-Man* aleatório, a presença da política de fuga ainda foi observada. A média de seleção da política permanece quase a mesma, quando aplicada comunicação a média diminui de $0,667$ para $0,600$. A seleção da política de busca também diminui e o de perseguição aumenta. Para o *Pac-Man* eficiente, a seleção da política de fuga, também é observada, e se mantém em $0,700$ independentemente da presença de comunicação, já as políticas de busca e perseguição funcionam de maneira inversa ao caso anterior, a de busca aumenta e a de perseguição diminui.

Tabela II. RESULTADO DOS EXPERIMENTOS COM DOIS AGENTES FANTASMAS

	RandomPacmanAgentTwo		*NimblePacmanAgent*	
Comunicação	Ausente	Presente	Ausente	Presente
Perseguição	0,533	0,733	0,667	0,533
Busca	0,800	0,667	0,667	0,767
Fuga	0,667	0,600	0,700	0,700
Indeterminado	0,000	0,000	0,033	0,000

Os outros quatro experimentos seguem o mesmo padrão dos experimentos anteriores, diferenciando apenas na quantidade de fantasmas, agora com três agentes. A Tabela III mostra o gráfico destes experimentos.

Tabela III. RESULTADO DOS EXPERIMENTOS COM TRÊS AGENTES FANTASMAS

	RandomPacmanAgentTwo		*NimblePacmanAgent*	
Comunicação	Ausente	Presente	Ausente	Presente
Perseguição	0,833	0,933	1,200	1,100
Busca	1,267	1,067	0,733	1,167
Fuga	0,900	1,000	1,067	0,633
Indeterminado	0,000	0,000	0,000	0,100

Novamente ainda há presença de fuga para ambos os agentes *Pac-Man*. Para o agente aleatório a quantidade aumenta quando há comunicação, de $0,900$ para $1,000$. No entanto, como o agente eficiente a fuga diminui de $1,067$ para $0,633$. Para as políticas de perseguição e busca os resultados são similares aos experimentos realizados com apenas dois fantasmas, com o *Pac-Man* eficiente, a busca novamente aumenta, e a perseguição diminui, com o *Pac-Man* aleatório, a busca diminui e a perseguição aumenta.

VI. CONCLUSÃO

Sistemas multiagentes distribuídos são capazes de multiplicar a eficiência comparado com um sistema de um único agente. No entanto, um desafio chave para este tipo de sistema é atingir cooperação a nível de grupo entre os agentes. A implementação de métodos de comunicação pode ser capaz de tratar tal desafio.

Este trabalho implementa dois tipos de comunicação direta com uma abordagem "quadro-negro", em múltiplos agentes autônomos, para analisar o impacto na aprendizagem de comportamentos colaborativos, adereçando dois problemas dos sistemas multiagentes distribuídos, a localidade e o ruído. O problema da localidade foi tratado utilizando a comunicação dos parâmetros de aprendizagem dos agentes. Já para adereçar o obstáculo do ruído, uma abordagem da comunicação da estimativa dos fantasmas sob a posição do *Pac-Man*, via mapas de probabilidade, foi implementada.

As implementações foram testadas por diversos experimentos, cada qual com trinta simulações. Seus resultados foram analisados:

- O erro proveniente do ruído diminui em 19,64% com a comunicação dos mapas de probabilidades, no entanto a diminuição do ruído não altera significativamente o número de instancias das simulações.
- A comunicação do aprendizado, quando o adversário é eficiente, aumenta as ocorrências das políticas de perseguição e busca e diminui as de fuga, porém, tal

comportamento não ocorre quando o adversário não é eficiente. Tal comunicação também demonstra impacto nas seleções de comportamentos nos experimentos, onde os agentes preferem utilizar políticas iguais ao invés de politicas complementares.

- A pontuação dos fantasmas não aumenta com a comunicação do aprendizado.
- A comunicação simultânea de ambos os métodos não gerou comportamentos colaborativos em todos os experimentos.

As politicas aprendidas pelos agentes com comunicação não geraram comportamentos cooperativos em todos os experimentos realizados. Portanto, é interessante analisar se outros tipos de comunicação são capazes de obter resultados melhores. Pode ser possível que ao realizar a união dos mapas de probabilidade em relação a posição dos fantasmas, os agentes tenham melhor conhecimento em relação a sua posição e a de seus aliados, podendo melhorar sua pontuação. Outro método de comunicação do aprendizado também pode ser implementado, pode ser que o compartilhamento da politica inteira, no fim do jogo, do agente que capturou a caça gere resultados melhores que a técnica implementada. Por fim, trabalhos futuros podem analisar o desempenho da comunicação em cenário de coevolução, onde a caça e os predadores evoluiriam simultaneamente.

AGRADECIMENTOS

O autor gostaria de agradecer o Prof. Dr. Guilherme Novaes Ramos, pelo voto de confiança ao fornecer-me a oportunidade de realizar esta pesquisa e por sua excelente orientação e suporte acadêmico durante todo o processo. Agradeço também a milha família pelo apoio nos momentos difíceis e aos meus amigos Khalil Carsten e Marcelo Araujo por acharem tempo para diversas leituras e fornecerem sugestões ao meu trabalho. E, por fim, agradeço ao Matheus Vieira Portela por desenvolver o excelente sistema utilizado no projeto.

REFERÊNCIAS

[1] Portela, Matheus V. “Seleção de comportamentos em múltiplos agentes autônomos com aprendizagem por reforço em ambientes estocásticos.” (2015). Trabalho de Graduação – Universidade de Brasília. Faculdade de Tecnologia.

[2] Portela, Matheus V. and Ramos, Guilherme N. “State estimation and reinforcement learning for behavior selection in stochastic multiagent systems.” SBC – Proceedings of SBGames 2015 | ISSN: 2179-2259

[3] Stone, Peter and Veloso, Manuela. “Multiagent systems: A survey from a machine learning perspective.” Autonomous Robots 8.3 (2000): 345-383.

[4] Balch, Tucker and Arkin, Ronald C. “Communication in reactive multiagent robotic systems.” Autonomous robots 1.1 (1994): 27-52.

[5] Stone, Peter and Veloso, Manuela. “Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork.” Artificial Intelligence 110.2 (1999): 241-273.

[6] Lebeltel, Olivier and Bessière, Pierre and Diard, Julien and Mazer, Emmanuel. “Bayesian robot programming” Autonomous Robots 16.1 (2004): 49-79.

[7] Russell, Stuart and Norvig, Peter “Artificial Intelligence: A modern approach.” Artificial Intelligence. Prentice-Hall, Englewood Cliffs 25 (1995): 27.

[8] Matarić, Maja J. “Using communication to reduce locality in distributed multiagent learning.” Journal of experimental & theoretical artificial intelligence 10.3 (1998): 357-369.

[9] Kaiser, M. and Dillman, R. and Rogalla, O. “Communication as the basis for learning in multi-agent systems.” ECAI’96 Workshop on Learning in Distributed AI Systems. 1996.

[10] Sutton, Richard S. and Andrew G. Barto. “Reinforcement learning: An introduction.” Vol. 1. No. 1. Cambridge: MIT press, 1998.

[11] Bussab, Wilton de O. and Pedro A. Morettin. “Estatística básica.” Saraiva, 2010.

[12] Thrun, Sebastian and Burgard, Wolfram and Fox, Dieter. “Probabilistic Robotics.” [S.l.]: MIT press, 2005.

[13] Weiss, Gerhard. “Multiagent systems: a modern approach to distributed artificial intelligence.” MIT press, 1999.

[14] Panait, Liviu, and Luke, Sean. “Cooperative multi-agent learning: The state of the art.” Autonomous agents and multi-agent systems 11.3 (2005): 387-434.

[15] Matarić, Maja J. “Reinforcement learning in the multi-robot domain.” Autonomous Robots 4.1 (1997): 73-83.

[16] Lucas, Simon M. “Computational intelligence and games: Challenges and opportunities.” International Journal of Automation and Computing 5.1 (2008): 45-57.

[17] Cesarino, Pedro S. “Implementação de Agente Racional Autônomo para Aprendizagem de Comportamentos Colaborativos.” Available: https://github.com/PedroSaman/Multiagent-RL

[18] Arai, Tamio, and Pagello, Enrico, and Parker, Lynne E. “Editorial: Advances in multi-robot systems.” IEEE Transactions on robotics and automation 18.5 (2002): 655-661.

[19] Koike, Carla M. C. E. C. “Bayesian Approach to Action Selection and Attention Focusing. An Application in Autonomous Robot Programming.” Diss. Institut National Polytechnique de Grenoble-INPG, 2005.

[20] Lebeltel, Olivier, and Bessière, Pierre, and Diard, Julien, and Mazer, Emmanuel. “Bayesian robot programming.” Autonomous Robots 16.1 (2004): 49-79.

[21] Tikhanoff, Vadim, and Cangelosi, Angelo, and Fitzpatrick, Paul, and Metta, Giorgio, and Natale, Lorenzo, and Nori, Francesco. “An open-source simulator for cognitive robotics research: the prototype of the iCub humanoid robot simulator.” Proceedings of the 8th workshop on performance metrics for intelligent systems. ACM, 2008.