Aprendizado pro Reforço Tabular Multiagente para Otimização de Filas

Pedro Henrique Gomes Mapa da Silva *

* *Departamento de Engenharia Elétrica, Universidade Federal de Minas Gerais, MG, (e-mail: pedrom072001@gmail.com).*

Abstract: In queues with a large flow of people who must perform various services to get out of the queue, it is interesting to optimize the layout and availability of resources, as well as movement strategies. This article studies the optimization of a hand washing queue in a restaurant using multi-agent Reinforcement Learning and with variable availability of actions. With results showing that collective strategies work better than selfish strategies and better configurations can be adopted to increase the flow of people and reduce the bottleneck.

Resumo: Em filas com grande fluxo de pessoas que devem realizar vários serviços para sair dela, é interessante a otimização da disposição e disponibilidade de recursos, além de estratégias de movimentação. Nesse artigo é estudado a otimização de uma fila para lavar as mãos em um restaurante utilizando Aprendizado por Reforço multiagente e com disponibilidade variável de ações. Com resultado mostrando que estratégias coletivas funcionam melhor que estratégias egoístas e melhores configurações podem ser adotadas para aumentar o fluxo de pessoas e diminuir o gargalo.

Keywords: Reinforcement Learning; Queue Optimization; Multi-agent Reinforcement Learning; Optimization of resource use; Organization of environments;

Palavras-chaves: Aprendizado por Reforço; Otimização de Filas; Aprendizado por Reforço multiagente; Otimização de uso de recursos; Organização de ambientes;

1. INTRODUÇÃO

A otimização de uma fila é útil para situações onde se tem vários parâmetros para operação de uma fila e não se sabe qual conjunto escolher, podendo ser um parâmetro, até mesmo, a estratégia das pessoas que chegam à fila. No caso desse trabalho, a fila estudada é a fila das pias para higienização das mãos do Restaurante Universitário I da UFMG, como na Figura 1. Nela, cada pessoa que entra na fila deve primeiramente passar pelos caixas e chegar a área de lavagem de mão, com várias pias, sendo que apenas algumas tem acesso a sabão e toalhas de papel, e, então, ela deve, em ordem, usar o sabão, então enxaguar as mãos e, por último, secá-las. Observando os frequentadores do restaurante, é possível ver que vários deles andam de um lado para o outro com o sabão na mão, buscando liberar espaço ou buscando se adiantar na busca de papel. Há também um recipiente de sabão móvel, que pode ser facilmente trocado de lugar.

Figura 1. Pias do RU I da UFMG

Para modelagem da fila e das pessoas que passam por ela, pode ser usado Aprendizado por Reforço, um dos principais paradigmas do Aprendizado de Máquina, onde um agente percebe seu ambiente, executa ações para alterar o ambiente e recebe recompensas sequencialmente, usando-as para aprender um comportamento ótimo, ou seja, um comportamento que maximiza a recompensa.

Esse trabalho busca traduzir o cenário da fila para o paradigma do Aprendizado por Reforço e estudar comportamentos e configurações que melhorem seu desempenho.

2. METODOLOGIA

Essa seção descreve como foi feita a tradução da fila para um modelo computacional compatível com o Aprendizado por Reforço.

2.1 Estados, ações e dinâmica

Primeiramente, foi preciso modelar a fila em termos de um ambiente que possa ter agentes, receba ações, mude de estado e retorne recompensas.

No caso da fila do restaurante, cada estado do ambiente possui 4 atributos, sendo eles posição, necessidade do agente, disponibilidade de cada uma das pias e quantidade de pessoas na fila. A posição pode ser cada uma das pias ou a zona de espera. A necessidade atual do agente pode ser sabão, esperar até terminar de esfregar o sabão, enxaguar, ou toalha de papel. A disponibilidade das pias é

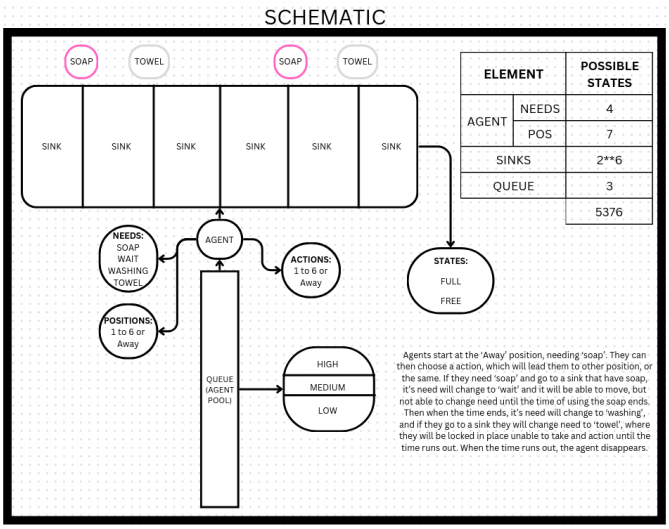

Figura 2. Modelo da fila para lavar as mãos

a informação de quais pias estão livres e quais não estão, sendo, no caso de 6 pias, um vetor binário de 6 posições. Já o número de agentes na fila é medido em 3 valores, 'LOW', 'MEDIUM' ou 'HIGH'.

Em cada um desses estados, o agente pode escolher 7 ações, que correspondem ao número de pias mais a posição da zona de espera, pois são os locais onde ele pode se deslocar, incluindo sua própria posição, ou seja, permanecer parado. No caso de uma pia estar ocupada, o espaço de ações será diminuído para conter apenas as posições livres.

A simulação é feita em passos individuas que consistem em, sequencialmente, selecionar os agentes que podem realizar ações, realizar essas ações, remover agentes que já terminaram os processos da fila, gerar novos agentes e, por fim, executar o algoritmo de aprendizado. Quando uma ação é tomada, caso um agente vá para uma pia que sacia sua necessidade atual, ele tem sua necessidade alterada para a próxima na ordem descrita, e um contador é adicionado a ela, fazendo ele ficar indisponível para tomar ações até que ela se encerre, exceto pela ações de esfregar o sabão, onde o agente ainda pode se mover.

Essa descrição do problema torná-o assíncrono e com número variável de ações, duas 1000 que expandem além do 1000 padrão no Aprendizado por Reforço. O esquemático dessa modelagem pode ser visto na Figura 2.

2.2 Algoritmo de aprendizado

Dentro do Aprendizado por Reforço, o aprendizado se dá associando cada par de estado e ação à um valor, chamado valor de ação, que indica o quão beneficial é escolher uma ação em um determinado estado, e, então, atualizando os valores de ação de acordo com as recompensas dadas durante a interação com o ambiente. Para fazer essa associação, podemos utilizar duas famílias de métodos principais, os métodos tabulares e os métodos de aproximadores de função. Os tabulares, assim como o nome indica, funcionam como uma tabela, os valores de ação são guardados em uma tabela com 3 colunas: os estados, as ações e os valores de ação, fazendo com que nenhum estado tenha associação com outro, todos estão bem separados em linhas diferentes. Já nos métodos de aproximação de funções, os valores de ação são codificados dentro de uma função que recebe como entrada um estado e uma ação e resulta em um valor de ação, onde essa codificação "mistura"a identidade de estados diferentes e associa-os, o que, se feito corretamente, pode acelerar o aprendizado.

Tanto no tabular quanto no caso de aproximação, a otimização é feita usando o algoritmo SARSA, atualiza o valor de ação associado a um estado e ação de acordo com a recompensa recebida e o valor de ação do próximo par de estados e ação.

2.3 Recompensa

Para recompensa, dois modos são propostos, o coletivista e o egocêntrico. Essa separação foi feita para testar qual objetivo de otimização teria a maior velocidade da fila e também porque a própria escolha da função de recompensa é sensível no aprendizado por reforço. No caso egocêntrico, cada agente tenta chegar ao fim do processo o mais rápido possível sem se importar com os outros, recebendo recompensa -1 para cada transição que não seja terminal, e um certo valor positivo para transição terminal. No caso coletivista, não há recompensa individual pela transição terminal individual, ao invés disso, há um sinal de recompensa que aumenta e diminui baseado em quantos agentes estiveram na fila recentemente, e todas transições recebem esse sinal, sendo que não há otimização usando SARSA para os passos terminais coletivistas, pois o objetivo é recompensar a velocidade da fila.

3. EXPERIMENTOS E ANÁLISE

Os experimentos foram divididos com base nos tipos de recompensa, coletivista ou egocêntrico, e tipo de método, tabular ou aproximação. O sinal de recompensa é usado para comparar diferentes políticas com o mesmo tipo de recompensa, mas o número de agentes médio também é usado, pois ele é um valor que deve ser minimizado e permite a comparação de até mesmo esquemas de recompensa diferentes.

Primeiramente, os experimentos foram feitos buscando comparar os tipos de recompensa e determinar qual o melhor regime de treinamento para minimizar o número de agentes na fila, depois, diferentes configurações de pias são comparadas para que a mais eficiente seja determinada. O repositório com o código está disponível no Github.

As pias foram configuradas da seguinte forma, são 6 pias, onde a primeira, a segunda e a terceira tem sabão, a segunda e a quarta tem toalhas e todas tem água disponível. O crescimento da fila foi de um nova agente a cada 3 passos da simulação, com um tamanho máximo de 10 agentes na zona de espera. As políticas utilizadas foram uma política aleatória, que seleciona ações aleatoriamente, e uma política ε-soft, que seleciona a ação com maior preferência com uma certa probabilidade e seleciona as ações restantes com uma chance menor, o que faz ela estar entre uma política aleatória e uma política *greedy*, que seleciona sempre a melhor ação.

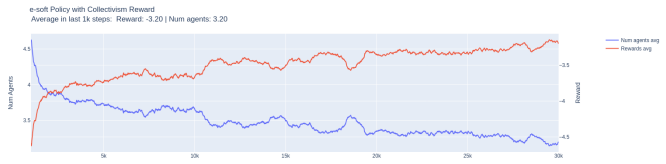

Figura 3. Resultados da Política ε-soft com Recompensa Coletivista

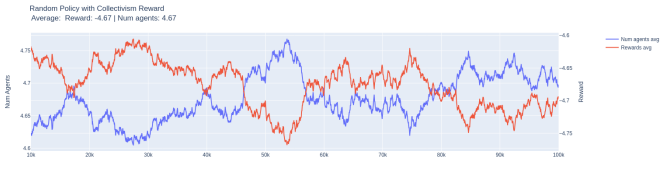

Figura 4. Resultados da Política Aleatória com Recompensa Coletivista

3.1 Tabular

Os experimentos abaixo foram feitos usando o método SARSA tabular.

Recompensa Coletivista No caso coletivista, a recompensa é definida como o negativo de uma média temporal do número de agentes na fila. Para o treinamento, o parâmetro de decaimento da recompensa foi de 0.8. SARSA foi configurado com $\alpha = 0.3$ and $\gamma = 0.9$. Já para a política ε-soft, ε foi de 0.4.

Os gráficos da evolução das políticas contendo o sinal de recompensa e o número de agentes em 30000 iterações do ambiente, usando uma média janelada das últimas 1000 iterações para facilitar a visualização, estão presentes nas Figuras 3 e 4. Primeiro, é interessante observar que a recompensa é próxima do número de agentes em módulo, independente da política, e, ainda, a correlação dos dois é -0.75, o que mostra que, quando o número total de agentes diminui, a recompensa aumenta, assim como desejado, ou seja, o sinal de recompensa codifica bem o resultado desejado. Finalmente, durante o treinamento, a recompensa vai subindo e, no nas últimas 1000 iterações foi de 3.2, bem mais baixo que 4.67 da política aleatória. Além disso, trocando o ε da política já treinada para 0, fazendo ela totalmente *greedy*, faz o número médio de agentes para 2.85.

Recompensa Egocêntrica No caso egocêntrico, a recompensa é -1 para toda interação, exceto para o estado terminal, que tem como recompensa 20. O restante das configurações foi mantido igual ao caso coletivista.

Os gráficos dos resultados, configurados assim como no caso coletivista, estão presentes nas Figuras 3 e 4. As recompensas e o número de agentes são muito menos parecidos, com a suavização não funcionando muito bem, pois há muito ruido na recompensa. Também, a correlação dos dois é -0.38, o que mostra que essa recompensa representa pior o objetivo de aumentar a velocidade da fila. Ao final do treinamento, o número de agentes médio das últimas 1000 iterações foi de 3.46, ou 3.11 fazendo a política treinada *greedy*. O resultado egocêntrico foi maior

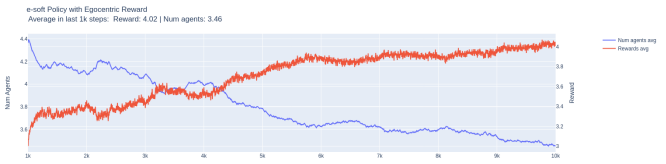

Figura 5. Resultados da Política ε-soft com Recompensa Egocêntrica

Figura 6. Resultados da Política Aleatória com Recompensa Egocêntrica

que o caso coletivista, mostrando que, se cada agente, ou cada pessoa na fila, tiver o objetivo de apenas sair da fila o mais rápido possível, o fluxo médio da fila será mais lento do que se houvesse uma estratégia coletiva.

3.2 Aproximação de funções

Os experimentos abaixo foram feitos usando o método SARSA com aproximação de funções, sendo ela implementada por uma rede neural com 1024 neurônios e ativação *tanh* sem regularização.

Para esse caso de aproximação com redes neurais, são introduzidos vários pontos que devemos ter atenção vindos da adição de redes neurais. Primeiramente, as entradas das redes devem ser adaptadas, sendo as categóricas formatadas na forma *one-hot*, e as outras como números reais com média em 0 e variância perto de 0 para acelerar convergência. Também, uma rede neural pode ser bem sensível aos seus hiper-parâmetros, como, por exemplo, a complexidade da rede, que não é ditada apenas pelo número de neurônios, mas, sim, também pelo algoritmo de otimização e pelas regularizações utilizadas. O *learning rate* deve ser baixo o suficiente para permitir que o algoritmo de semi-gradiente funcione, mas grande o suficiente para o algoritmo convergir em tempo hábil. A exploração inicial não pode ser garantida apenas pela rede, pois ela pode não promove, como no caso tabular, exploração por valores iniciais, então a exploração deve vir de outra forma, como um ε maior. Uma adição ao algoritmo foi a queda do ε para que ele gradualmente se aproximasse de uma política *greedy*.

Como esperado, pela variabilidade das redes, o treinamento também teve ainda mais variabilidade, quando comparado com o caso tabular, mas, depois de ajustar o *learning rate* para 0.01 e o ε para decair lento e exponencialmente, tivemos convergência assim como para o caso tabular, com variação de menos de 10% no resultado final e no tempo de convergência.

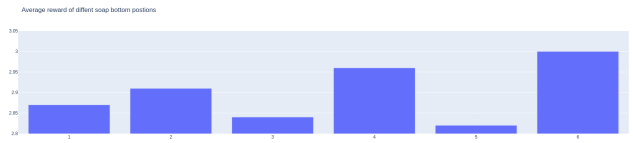

Figura 7. Resultados de diferentes posicionamentos da saboneteira

3.3 Configurações de pias

Agora, usando a configuração real das pias, como na Figura 1, consideramos a configuração ideal da saboneteira móvel disponível que pode ser deslocada facilmente e consegue alimentar uma pia com sabão. A Figura 7 abaixo mostra o número de agentes médio da versão *greedy* das políticas treinadas para cada posicionamento da saboneteira. As políticas foram treinadas com recompensa coletivista e seguindo as configurações descritas na seção da mesma. No caso, o melhor posicionamento foi na pia de número 5, que tinha apenas toalhas de papel disponíveis.

4. CONCLUSÃO

Foram treinados agentes com Aprendizado por Reforço num cenário multiagente e com disponibilidade variável de ações para encontrarem a melhor política e a melhor configuração de pias que maximizasse o fluxo da fila em dois cenários, um onde os agentes se importam apenas com si mesmo, e outro onde eles buscam maximizar o fluxo da fila. A maior velocidade da fila foi no último caso, chamado de coletivista, que ficou bem abaixo do outro cenário egocêntrico e de uma política aleatória testada como *baseline*. Também foi testado o posicionamento de uma saboneteira, em cada uma das pias, mostrando que a quinta pia seria o melhor lugar para botar uma nova saboneteira.