Uma Abordagem utilizando Redes Neurais *Fuzzy* ART e Aprendizagem por Reforço para o Problema dos k-servos.

Mademerson Leandro da Costa[1], Carlos Alberto de Araújo Padilha[2], Jorge Dantas de Melo[2] e Adrião Duarte Dória Neto[2]

[1]Departamento de Matemática e Estatística, Universidade do Estado do Rio Grande do Norte

[2]Laboratório de Sistemas Inteligentes, Departamento de Computação e Automação, Universidade Federal do Rio Grande do Norte

Campus Universitário s/n – 59072-970 – Natal – RN – Brasil

Endereço para Correspondência

E-mails: mademersonleandro@uern.br, carlosalberto@dca.ufrn.br, adriao@dca.ufrn.br, jdmelo@dca.ufrn.br,

Resumo – Este artigo propõe um método de decisão em múltiplas etapas com a utilização de Redes Neurais e a aplicação de técnicas de Aprendizagem por Reforço para o problema do k-servos (PKS). Para isso, foi utilizada a Rede Neural *Fuzzy* ART e o algoritmo de clusterização *k-means* buscando reduzir a dimensão do problema e, em seguida, foi aplicando o algoritmo *Q-Learning* para sua solução e comparado seu desempenho para os dois métodos de agrupamentos propostos.

Palavras-chave: aprendizagem por reforço, problema dos k-servos, redes neurais, *fuzzy* ART.

1. Introdução

O problema dos *k*-servos (PKS), introduzido por Manasse, McGeoch e Sleator [1], é um problema *online* com amplas possibilidades de aplicação e bastante estudado na literatura. O mesmo pode ser definido da seguinte forma: dada uma configuração inicial de servos em um grafo G e uma sequência de requisições σ, encontre uma política π que indique os movimentos dos servos de modo que o custo total dos deslocamentos seja o menor possível. Como o atendimento de cada uma das requisições está associado ao deslocamento de um servo, frequentemente, o objetivo a ser alcançado é a minimização da distância total percorrida por todos os servos. O PKS pode ser visto como um processo de decisão em múltiplas etapas, onde a cada instante t_i (i = 1, 2, . . . , m) considerado, uma decisão deve ser tomada sobre como atender à solicitação σ_i. Esse processo é também *markoviano*, uma vez que a decisão a ser tomada no instante t_i depende apenas das informações disponíveis nesse instante. Dentre as metodologias utilizadas na solução dos problemas de decisão *markovianos*, uma abordagem bastante eficiente é a aprendizagem por reforço [2]. O objetivo buscado é determinar uma política ótima para a tomada de decisões. Para isso, busca-se treinar um agente que implemente a política ótima tendo como base sua experiência anterior. Ao agente são apresentadas situações ou percepções de seu ambiente (configurações dos servos, no caso do PKS), representadas por estados, aos quais o agente responde com ações (movimentos dos servos para atender um determinado nó demanda). Para cada ação tomada, uma avaliação de seu resultado, na forma de recompensa ou punição, é apresentada ao agente pelo ambiente. O processo de aprendizagem tem por finalidade orientar o agente a tomar as ações que venham a maximizar (minimizar) as recompensas (punições) recebidas. Deve-se levar em conta que uma ação tomada em um dado instante influencia não apenas a avaliação imediata, mas, também, todas as outras ações que serão efetuadas a partir de então. Trata-se, portanto, do problema de como mapear estados do ambiente em ações do agente de forma a maximizar (minimizar) um dado retorno. Entretanto, a dimensão da estrutura de armazenamento utilizada pela aprendizagem por reforço para se obter a política ótima cresce em função do número de estados e de ações, que por sua vez é proporcional ao número *n* de nós e *k* de servos. Ao se analisar esse crescimento (matematicamente, $C_{n,k} \approx \mathcal{O}(n^k)$) percebe-se que o mesmo ocorre de maneira exponencial, limitando a aplicação do método a problemas de menor porte, onde o número de nós e de servos é reduzido. Este problema, denominado maldição da dimensionalidade, foi introduzido por Belmann [3], e resulta na impossibilidade de execução de um algoritmo para certas instâncias de um problema pelo esgotamento de recursos

computacionais para obtenção de sua saída. De modo a evitar que a solução proposta baseada exclusivamente na aprendizagem por reforço seja restrita a aplicações de menor porte, propõe-se uma solução alternativa para problemas mais realistas, que envolvam um número maior de nós e de servos. Esta solução alternativa é executada em múltiplas etapas: inicialmente o total de nós são divididos em subgrupos utilizando uma rede neural *fuzzy* ART, em seguida, para a solução do PKS foi utilizada a aprendizagem por reforço, aplicada a um número reduzido de nós obtidos a partir do processo de agregação, e um método guloso, aplicado aos subconjuntos de nós resultantes do processo de agregação, onde o critério de escolha do agendamento dos servos é baseado na menor distância ao local de demanda.

Este artigo está organizado como segue: na seção 2 há um breve referencial teórico, abordando conceitos sobre redes *fuzzy* ART e aprendizagem por reforço. Na seção 3 mostramos com mais detalhes o método proposto. Na seção 4 mostramos os resultados dos testes realizados. Na seção 5 apresentamos as conclusões e trabalhos futuros.

2. **Referencial teórico**

2.1 O Problema do *k*-servos

O problema dos k-servos *online* foi introduzido por Manasse, McGeoch e Sleator [1]. Este modelo proposto provê uma abstração para um grande número de problemas. Mais ainda, a conjectura e o modelo dos k-servos têm proporcionado um significativo desenvolvi mento da análise competitiva [4]. O problema dos k-servos é bem simples de ser descrito. Denominam-se servos as unidades de atendimentos das demandas. Demanda é a indicação do local onde os atendimentos deverão ser realizados, ou seja, os servos devem atender às solicitações originadas nestes locais de demanda. Considere um espaço métrico M habitado por k servos localizados em pontos de M não necessariamente distintos. Inicialmente, cada servo é posicionado em algum ponto de M. No decorrer do tempo, pedidos por serviço chegam de algum ponto do espaço métrico. A meta é servir a sequência de pedidos. Um pedido j é simplesmente um ponto do espaço métrico, e servir j implica em deslocar um servo ao ponto j. Associado ao deslocamento do servo localizado em i para um local j existe um custo de atendimento proporcional à distância d(i, j) do servo em i à demanda em j. O objetivo é minimizar a distância total percorrida pelos servos no atendimento de todas as requisições (vetor-demanda). Portanto, um algoritmo que resolva o problema dos k-servos de maneira *online* deve decidir qual servo será deslocado para atender a cada uma das requisições, de modo que a distância total percorrida pelos mesmos seja a menor possível. Vários algoritmos são capazes de solucionar o problema dos k-servos. Entre eles o algoritmo de Sleator e Tarjan [1], a estratégia gulosa [5], além de soluções mais elaboradas cujas taxas de competitividade já foram estabelecidas, como os algoritmos *Harmonic* e *Work Function* [4]. Bansal *et al*. [6] utilizando um algoritmo poli-logarítimíco competitivo aleatório para o problema dos k-servos em um espaço métrico finito arbitrário obteve uma taxa de competitividade de $(\tilde{O}log^3 nlog^2 k)$. Rudec *et al*. [7] demonstra empiricamente que devido a sua alta complexidade computacional o WFA funciona de forma semelhante ou apenas ligeiramente melhor que uma heurística simples, tal como o algoritmo guloso, para algumas instâncias do problema dos k-servos *on-line*.

2.2 Redes Neurais *Fuzzy* ART

O principal desafio para o agrupamento de sobreposição de dados é a determinação do número adequado de *clusters* e de divisão da região de sobreposição [8]. Um modelo *Fuzzy Adaptive Resonance Theory* (*Fuzzy*ART) [9] é capaz de aprender de forma estável e rápida o reconhecimento categorias em resposta as entradas de sequências arbitrárias de padrões analógicos ou binários. A *Fuzzy* ART incorpora os cálculos da teoria dos conjuntos *fuzzy* para a rede neural ART 1, que aprende a categorizar apenas padrões de entrada binários. A generalização da aprendizagem para padrões de entrada analógicos e binários é atingido através da substituição do operador de interseção ($\cap$) em uma ART 1 pelo operador MIN (Λ) da teoria dos conjuntos *fuzzy*. O operador MIN é reduzido ao operador de interseção no caso binário. A proliferação de categorias é impedida pela normalização dos vetores de entrada em um estágio de pré-processamento. Um procedimento de normalização chamado de codificação complementar leva a uma teoria simétrica em que o operador M1N (Λ) e o operador MAX (V) da teoria dos conjuntos *fuzzy* atuam em papéis complementares. A codificação complementar utiliza células *on* e *off* para representar o padrão de entrada, e preserva amplitudes de recursos individuais enquanto normalizar o vetor total de células *on/off*. A aprendizagem é estável, pois todos os pesos adaptativos só podem diminuir com o tempo. Pesos decrescentes correspondem a tamanhos crescentes de categorias "caixas". Valores de vigilância menores levam a maiores “caixas” de categoria. A aprendizagem pára quando o espaço de entrada é coberta por “caixas”. Com o aprendizado rápido e um

conjunto finito de entrada de tamanho e composição arbitrários, a aprendizagem estabiliza depois de apenas uma apresentação de cada padrão de entrada. Uma opção de entrega rápida com recodificação lenta combina aprendizagem rápida com o esquecimento de regras que sistemas de *buffers* de memória contra o ruído. Usando esta opção, eventos raros podem ser rapidamente aprendidos, ainda memórias anteriormente aprendidas não são rapidamente apagadas em resposta a flutuações de entrada estatisticamente confiáveis.

2.3 Algoritmo *k-means*

O *K-means* [10] é um dos mais conhecidos e um dos mais simples algoritmos de aprendizagem não supervisionada para a resolução de problemas de agrupamentos. Consiste em um procedimento simples cujo objetivo é classificar um determinado conjunto de dados através de um certo número k de clusters (aglomerados) fixado *a priori*. A ideia principal é definir k centroides, um para cada cluster. Estes centroides devem ser colocados de uma forma astuta, pois diferentes localizações resultam em diferentes resultados. Então, a melhor opção é colocá-los o mais longe possível um do outro. O próximo passo é levar cada ponto pertencente a um determinado conjunto de dados e associá-lo ao centroide mais próximo. Quando nenhum ponto está pendente, o primeiro passo está concluído e um agrupamento inicial está feito. Neste ponto, precisamos recalcular k novos centroides como baricentros dos clusters resultantes da etapa anterior. Após termos estes k centroides novos, uma nova ligação tem de ser feita entre os mesmos pontos de dados e definir o mais próximo centroide novo, gerando assim uma nova etapa. Como resultado desta etapa, podemos notar que os k centroides mudam a sua localização passo a passo até que não haja mais mudanças para serem feitas. Em outras palavras centroides não se movem mais.

2.4 Aprendizagem por Reforço

A aprendizagem por reforço (do inglês, *reinforcement learning*) é um tema multidisciplinar, envolvend1o disciplinas como Biologia, Ciências da Computação, Ciências Cognitivas, Engenharia, Filosofia, Física, Matemática, Psicologia e Sociologia. Trata-se de um ramo do aprendizado de máquinas que estuda o aprendizado a partir da interação entre agente e seu ambiente. Um agente aprendiz é aquele que, a partir da interação com o ambiente que o cerca, aprende de maneira autônoma uma política ótima de atuação por experimentação direta, sem ser ensinado por meio de exemplos fornecidos por um supervisor. O ambiente é todo o sistema físico externo que irá interagir com o agente, percebido por este através de um conjunto de informações denominado de estado do ambiente. Pode-se dizer que a aprendizagem por reforço é uma forma de aprendizado não-supervisionado que ensina como mapear estados para ações, de modo a maximizar (minimizar) um sinal numérico de retorno, que representa a soma total de todas as recompensas (punições) decorrentes das ações tomadas. Não são ditas ao agente de aprendizagem quais ações devem ser tomadas, assim como ocorre em outras abordagens. O agente deve descobrir quais ações resultam nas maiores recompensas (menores punições), a partir das percepções do ambiente, representadas por estados, que lhe são apresentadas. Para cada ação tomada, uma avaliação do seu resultado, na forma de recompensa ou punição, é apresentada ao agente pelo ambiente. O processo de aprendizagem tem por finalidade orientar o agente a tomar as ações que venham a maximizar (minimizar) as recompensas (punições) recebidas.

Figura 1: Diagrama esquemático de um sistema de aprendizagem por reforço.

Q-Learning

Um dos métodos de resolução mais importantes da aprendizagem por reforço é o algoritmo *Q-Learning* [11]. Tal algoritmo é baseado nos conceitos do método de diferenças temporais e sua convergência para os valores ótimos de Q (Q*(s, a)) independe da política que está sendo utilizada. A expressão de atualização do valor de Q do algoritmo *Q-Learning* é a seguinte:

$$Q(s, a) \leftarrow Q(s, a) + \alpha.[r_{t+1} + \gamma \max_{a'} Q(s', a') - Q(s,a)]$$

onde r_{t+1} é o retorno associado à transição do estado s para o estado s', α é a taxa de aprendizagem e γ é o fator de desconto, com $0 \leq \lambda \leq 1$. A função de valor do estado atual (Q(s, a)) é atualizada a partir do seu valor atual, do reforço imediato (r_{t+1}) e da diferença entre a máxima função de valor no estado seguinte ($\max_{a'} Q(s', a')$) e o valor da função de valor do estado atual. Uma característica marcante do *Q-Learning* é que a função de valor Q aprendida aproxima-se diretamente da função de valor ótimo Q^*, sem depender da política que está sendo utilizada.

Este fato simplifica a análise do algoritmo. A política ainda mantém algum efeito ao determinar qual dos pares estado-ação deve-se visitar e atualizar. A convergência exige que todos os pares estado-ação sejam visitados. Logo, a política π a ser utilizada para a determinação do Q^* deve ser ε-gulosas, pelo menos. O algoritmo *Q-Learning* é apresentado abaixo:

Inicializar Q(s, a) de forma arbitrária

Repita (para cada episódio)

Inicializar s;

Repita (para cada passo do episódio);

Escolher *a* para *s* usando uma política π (ε-gulosa, por ex.);

Tomar a ação *a*, observar r, s';

$Q(s, a) \leftarrow Q(s, a) + \alpha.[r_{t+1} + \gamma max_{a'}.Q(s', a') - Q(s,a)];$

s ← s';

até *s* ser o estado final;

Fim

Figura 2: Algoritmo *Q-learning*.

3 Método proposto

O método possui duas fases, uma de agrupamento e outra de aprendizagem. Na fase de agrupamento, dividimos os nós do grafo em grupos e encontramos os nós representantes dos mesmos. Para isso, utilizamos a rede *Fuzzy* ART para separação dos agrupamentos. Já que na Fuzzy ART o número de grupos não precisa ser determinado *a priori*, sendo controlado apenas pelo parâmetro de vigilância ρ. Foi utilizado ainda com o proposito de comparação o algoritmo de agrupamento *k-means*. Já na fase de aprendizagem, utilizamos a aprendizagem por reforço em um número reduzido de nós, contornando o problema da maldição da dimensionalidade.

Dado um grafo, normalizamos as coordenadas dos nós porque a rede *Fuzzy* ART só aceita pontos que tenham coordenadas entre 0 e 1. Em seguida, usamos os pontos normalizados para treinar a rede *Fuzzy* ART utilizando um determinado valor para o parâmetro de vigilância (parâmetro que controla o número de categorias ou grupos) e conseguimos que os pontos fiquem distribuídos em *n* grupos. Para o *k-means* foi fornecido o mesmo conjunto de pontos e a quantidade *clusters* da rede *Fuzzy* ART.

De posse dos grupos, partimos para a fase de aprendizagem, onde o algoritmo *Q-Learning* será utilizado para treinar os *k* servos para encontrar a política ótima de deslocamento dentro de cada grupo e entre os grupos, onde cada grupo será representado pelo seu respectivo nó-centro. Entretanto, nem todos os pares estado-ação que compõem o problema original vão ser visitados. Por isso, foram feitas as seguintes considerações como em [12]:

1. Os 2 servos estão localizados em nós-centro distintos e surge uma demanda em um local, que não é nó-centro, num grupo distinto ao dos servos: o algoritmo considerará que a demanda se encontra no nó-central de seu grupo. Assim, como este par estado-ação já foi visitado, implica que a seleção do servo se dará pelo maior valor da política π correspondente ao estado-ação.
2. Os 2 servos estão localizados em grupos distintos e surge uma demanda num nó que pertence ao grupo de um deles: o servo a ser deslocado será aquele que pertencer ao grupo de um deles. Se por acaso fossem 3 servos, e 2 deles pertencessem ao grupo da demanda e 1 não, o deslocado seria aquele pertencesse ao grupo da demanda e que estivesse mais próximo a mesma (critério guloso).
3. Os 2 servos estão localizados em um mesmo grupo: independente de onde estiverem localizados os servos e a requisição por serviço, o servo a ser deslocado será escolhido pelo método guloso.
4. Os 2 servos estão em grupos distintos, um deles está num nó centro e outro não, e surge uma demanda em um nó que não é centro e que pertence a um grupo que é distinto ao dos 2 servos: tanto o servo quanto a demanda serão consideradas como se estivessem localizados no nó-central do seu grupo. Este estado transposto já foi visitado durante o aprendizado e o servo escolhido será o que apresentar o maior valor para a política π correspondente.

```
Dado um grafo com N nós

Normalize as coordenadas dos N nós entre 0 e 1;

Treine uma rede Fuzzy ART com as coordenadas dos N nós e encontre n grupamentos;

Para cada grupamento x formado

    Execute o Q-Learning no conjunto de nós que compõem o grupo;

    Selecione o nó que será o centro do grupo;

Fim

Execute o Q-Learning nos x nós-centro selecionados e com k servos e encontre a política π;

Para cada demanda σ_i

    Se o par estado-ação foi visitado pelo Q-Learning

       O servo a ser deslocado será determinado pela política π;

    Senão Teste a consideração 2;

    Senão Teste a consideração 3;

    Senão Teste a consideração 1;

    Senão Teste a consideração 4;

Fim do algoritmo
```

Figura 3: Método Proposto

4 Resultados

Nesta seção serão apresentados os resultados experimentais obtidos neste trabalho. Todos os testes foram feitos no ambiente MATLAB.

Em [12] é feito um teste comparativo entre o *Q-Learning* e a estratégia hierárquica, que mostra que o desempenho do algoritmo hierárquico (dividindo o grafo em regiões e em nós centros com o uso do algoritmo de Boruvka) não é expressivamente inferior ao do *Q-Learning*, por isso não será incluído tal teste comparativo aqui.

O treinamento da rede *Fuzzy* ART foi feito utilizando o parâmetro de vigilância em 0.2, controlando a proliferação de grupos. Já o treinamento do *Q-Learning* para cada grupo e para os nós-centros foi feito a partir de uma sequência de 100.000 episódios, considerando $\alpha = 0.85, \gamma = 0.2$ e $\epsilon = 0.1$. Os dados registrados nos testes são mostrados na Tabela 1, onde fazemos a comparação do desempenho do método proposto e do algoritmo guloso para 10^6 demandas.

No teste 1, geramos um grafo com 50 nós e 2 servos para atender as demandas. A rede *Fuzzy* ART agrupou os 50 nós em 4 grupos, assim podemos calcular o percentual de redução de pares estado-ação conseguido:

$$C_{50,2} \cdot 50 \times 2 = 122500\ pares$$

$$C_{4,2} \cdot 4 \times 2 = 48\ pares$$

$$Redução\ de\ 99{,}96\%$$

No teste 2, geramos um grafo com 100 nós e 2 servos para atender as demandas. A rede *Fuzzy* ART agrupou os 100 nós em 4 grupos, assim podemos calcular o percentual de redução de pares estado-ação conseguido:

$$C_{100,2} \cdot 100 \times 2 = 990000\ pares$$

$$C_{4,2} \cdot 4 \times 2 = 48\ pares$$

$$Redução\ de\ 99{,}9952\%$$

No teste 3, geramos um grafo com 100 nós e 2 servos para atender as demandas. Foi utilizado o algoritmo *k-means* para divisão dos grupos. Neste método, o número de grupos deve ser estabelecido antecipadamente. Desta forma, fica a cargo do pesquisador estabelecer a quantidade de grupos, o que pode implicar numa menor redução na dimensão do problema. Sabendo disso, resolvemos fixar a mesma quantidade de grupos gerados pela rede *Fuzzy AR*T. Assim, o objetivo da utilização do algoritmo *k-means* é apenas comparar o *Q-learning* com outro método de agrupamento e testar sua eficiência.

Teste 1		
Algoritmos	Proposto	Guloso
Menor caminho	0*	0*
Maior caminho	1.04	1.0
Média	0.37	0.40
Desvio	0.2198	0.2402
Vitórias (%)	76.34%	23.66%
Teste 2		
Algoritmos	Proposto	Guloso
Menor Caminho	0*	0*
Maior Caminho	1.14	1.14
Média	0.40	0.61
Desvio	0.23	0.30
Vitórias (%)	87.4%	12.6%
Teste 3		
Algoritmos	Proposto	Guloso
Menor Caminho	0*	0*
Maior Caminho	1.18	0.95
Média	0.46	0.5
Desvio	0.24	0.22
Vitórias (%)	63.5%	36.5%

*Durante a geração aleatória de demandas, uma demanda apareceu onde estava localizado um dos servos.

Conclusão

O método visa a aplicação da aprendizagem por reforço ao problema do k-servos em instâncias de maior dimensão. Para isso, buscou reduzir a complexidade do problema através da divisão do problema em subproblemas e assim conseguindo um grande ganho no desempenho, como mostrado nos resultados com redução de até 99,99% no número de pares estado-ação. Além disso, o método se mostrou mais eficiente se comparado com a estratégia gulosa, na média percorreu menores distâncias.

Referências Bibliográficas

[1] Manasse, M. S., McGeoch, L. A. e Sleator, D. D., *Competitive Algorithms for On-Line Problems*, Proc. 20th Annual ACM Symposium on Theory of Computing, pags. 322-33, 1988.

[2] Sutton, R. S. e Barto, A. G., *Reinforcement Learning: An Introduction*, The MIT Press, 1998.

[3] Bellman, R., *Dynamic Programming*, Princeton University Press, 1957.

[4] Borodin, A. e El-Yaniv, R., *Online Computation and Competitive Analysis*, Cambridge University Press, 1998.

[5] Goldbarg, M. C. e Luna, H. P. C., Otimização Combinatória e Programação Linear: Modelos e Algoritmos, Editora Campus, 2000.

[6] Bansal, N.; Buchbinder, N.; Madry, A.; Naor, J. *A Polylogarithmic-Competitive Algorithm for the k-Server Problem*. Foundations of Computer Science (FOCS), 52nd Annual Symposium on IEEE, 2011.

[7] Rudec, T.; Baumgartner, A.; Manger, R. *Measuring true performance of the work function algorithm for solving the on-line k-server problem*. Information Technology Interfaces (ITI), on 32nd International Conference, 2010.

[8] Lee Onn Mak; Gee Wah Ng ; Lim, G. ; Kezhi Mao. *A merging Fuzzy ART clustering algorithm for overlapping data*. Foundations of Computational Intelligence (FOCI), Symposium on IEEE, 2011.

[9] Gail A. Carpenter, Stephen Grossberg, and David B. Rosen, *Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System*. Neural Networks, Vol. 4, pp. 759-771, 1991.

[10] Macqueen, J., *Some methods for classification and analysis of multivariate observations*. In 5-th Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[11] Watkins, C. J. C. H., *Learning from Delayed Rewards*, Phd thesis, University of Cambridge, 1989.

[12] Lima Júnior, M. L., Melo, J. D. de e Dória Neto, A. D., Uma Contribuição à Solução do Problema dos *k*-Servos Usando Aprendizagem por Reforço, Dissertação de Mestrado, Universidade Federal do Rio Grande do Norte, UFRN, Natal, 2005.