Estratégias de Atualização de Políticas para a Coordenação de Agentes Baseados em Enxames[1]

Richardson Ribeiro[2]

Fabrício Enembreck[3]

Resumo. Neste artigo é analisada a influência dos parâmetros de aprendizagem de algoritmos baseados em enxames e propostas estratégias para a atualização de políticas em ambientes dinâmicos. Nós verificamos que os parâmetros de aprendizagem quando ajustados inadequadamente podem ocasionar atrasos no processo de aprendizagem ou convergir para uma política não-satisfatória. Além disso, esse problema é agravado em ambientes dinâmicos, pois o ajuste dos parâmetros de algoritmos que utilizam recompensas não é suficiente para garantir uma boa convergência. Para tal problema, nós desenvolvemos estratégias de atualização de políticas que modificam os valores das recompensas, melhorando a coordenação dos agentes que atuam em ambientes dinâmicos. Para isso, foi desenvolvido um *framework* capaz de demonstrar de maneira iterativa a influência dos parâmetros e das estratégias de atualizações. Resultados experimentais mostram que é possível acelerar a convergência para uma política global consistente, superando os resultados de abordagens clássicas de algoritmos baseados em enxames.

---

[1] Este trabalho foi parcialmente publicado no Simpósio de Computação Aplicada (SCA'09) e foi selecionado como um dos três melhores artigos do evento.

[2] Universidade Tecnológica Federal do Paraná (UTFPR), Pato Branco - Pr, Brasil.
{richardsonr@utfpr.edu.br}

[3] Pontifícia Universidade Católica do Paraná (PUCPR). Programa de Pós-Graduação em Informática (PPGIa), Curitiba - Pr, Brasil.
{fabricio@ppgia.pucpr.br}

Abstract. This paper analyzes the influence of the learning parameters of swarm intelligence algorithms and proposes strategies to updating policies generated by rewards for dynamic environments. We observed that when the parameters of algorithms based on rewards are set improperly delays in the learning and convergence to a non-satisfactory solution may occur. Moreover, this problem is aggravated in dynamic environments, because the parameter setting is not sufficient to ensure convergence. To address such problem, we developed strategies that modify the pheromone values, improving the coordination among the agents and letting convergence even in dynamic environments'. For such, a framework able to demonstrate in an iterative way the influence of parameters and strategies was developed. Experimental results show that it is possible to accelerate the convergence to a consistent global policy, overcoming the results of classic swarm intelligence algorithms.

1 Introdução

A coordenação entre agentes, quando bem aplicada, pode contribuir para a execução eficiente de tarefas complexas. A coordenação das ações dos agentes pode ajudar a evitar problemas como: soluções redundantes para um subproblema; inconsistência de execução (atuação sobre subproblemas obsoletos); desperdício de recursos e *deadlocks* (espera por eventos que provavelmente não irão ocorrer) [23].

Há importantes aplicações no mundo real onde a coordenação é necessária, por exemplo, gerenciamento de tráfego, redes de sensores [2], gerenciamento de cadeias de suprimentos, monitoramento ambiental, modelagem estrutural, gerenciamento de desastres e manufatura. Nessas aplicações, os agentes atuam em ambientes incertos e dinâmicos, sendo o controle realizado de forma autônoma pelos próprios agentes utilizando apenas suas percepções locais.

Em tais aplicações, os agentes precisam selecionar ações e se relacionar com os demais agentes, baseado no conhecimento do ambiente, limitações de recursos e restrições de comunicação. Nesses casos, métodos de coordenação devem ser utilizados no gerenciamento das dependências que ocorrem quando agentes possuem objetivos inter-relacionados, ou seja, quando os agentes compartilham um ambiente ou quando compartilham recursos.

O paradigma de coordenação baseado em inteligência de enxames (*swarm intelligence*) tem sido estudado intensamente por diferentes pesquisadores [3] [13] [24]. O paradigma é

inspirado nas colônias de insetos sociais, onde sistemas computacionais reproduzem seu comportamento para a resolução de problemas coletivos, como por exemplo, colônias de formigas, abelhas, cupins ou vespas. Tais colônias possuem características desejáveis (adaptação e coordenação), para soluções de problemas computacionais que necessitam de coordenação. Pesquisas anteriores sobre a organização de colônias de insetos sociais e suas aplicações na organização de sistemas multiagente mostram bons resultados em problemas complexos, como a otimização combinatória [4].

No entanto, uma das principais dificuldades desses algoritmos é o tempo de convergência que pode ser bastante lento para muitas aplicações do mundo real. Nestes problemas, não há garantia de convergência de algoritmos baseados em recompensas, pois é sabidamente conhecido que tais algoritmos foram inicialmente desenvolvidos e aplicados a problemas estáticos, onde a função-objetivo não altera ao longo do tempo. No entanto, raramente há problemas do mundo real que são estáticos, ou seja, problemas em que não ocorrem mudanças de prioridades por recursos, mudanças nos objetivos e tarefas que não são mais necessárias. Estas particularidades caracterizam um ambiente dinâmico.

O uso de sistemas baseados em insetos, em especial algoritmos de colônia de formigas, tem atraído a atenção de pesquisadores por reproduzir sofisticadas estratégias de exploração e apresentar generalidade e robustez [4]. No entanto, na maioria das vezes estes sistemas não são capazes de melhorar a coordenação dos agentes em ambientes dinâmicos, devido à necessidade de adequar os parâmetros de aprendizagem às alterações do ambiente.

Nós analisamos os principais parâmetros do algoritmo *Ant-Q* [7] e discutimos em [16] o impacto da variação dos parâmetros na convergência dos agentes. Um *framework* foi desenvolvido para demonstrar de maneira iterativa a influência dos parâmetros enquanto agentes deliberam no sistema. Neste trabalho, nós estendemos o trabalho apresentado em [16] propondo a aplicação de estratégias de atualização de recompensas para ambientes dinâmicos. Nós reutilizamos neste trabalho alguns princípios apresentados em [18], onde foi discutida a importância dos parâmetros em algoritmos de aprendizagem por reforço, analisando o impacto dos parâmetros do algoritmo em problemas de congestionamento de tráfego urbano.

Um aspecto importante é investigar se políticas podem ser reutilizadas para encontrar rapidamente uma solução após modificações no ambiente. As estratégias propostas neste trabalho reagem em função das mudanças no ambiente. Podemos citar como exemplo de alteração dinâmica, a movimentação física de um estado em um ambiente (*i.e.* a alteração cartesiana de uma cidade no problema do caixeiro viajante). Tais estratégias são baseadas em recompensas (feromônios) de políticas passadas, utilizadas para guiar os agentes a novas soluções.

Este artigo está organizado da seguinte maneira: na Seção 2 são descritas algumas abordagens relacionadas à otimização e algoritmos de colônia de formigas. Já na Seção 3 são descritas as estratégias de atualização de políticas para algoritmos baseado em colônias de formigas. A Seção 4 apresenta o *framework* utilizado para avaliar o impacto dos parâmetros

do algoritmo *Ant-Q*. Os resultados experimentais são mostrados na Seção 5 e finalmente na Seção 6 conclusões e perspectivas para trabalhos futuros são discutidas.

2 Otimização e Algoritmos de Colônia de Formigas

A otimização por colônia de formigas (*Ant Colony Optimization*) [3] é uma abordagem bem sucedida baseada em população e aplicada em problemas de otimização combinatória. Nesses problemas, só é possível encontrar a solução ótima se o espaço de estados for explorado completamente, aumentando exponencialmente o custo computacional à medida que o espaço de estados aumenta.

Algoritmos de colônia de formigas, como *Ant System* [3], *Ant Colony System* [4] e *Ant-Q* [7] têm sido aplicados com algum sucesso em problemas de otimização como: caixeiro viajante [3], coloração de grafos [1], roteamento de veículos [8], entre outros. Tais algoritmos são baseados no comportamento forrageiro das formigas (agentes), que seguem um padrão de decisão baseado na aleatoriedade [5]. À medida que o objetivo é localizado (*e.g.*, fonte de alimento), as formigas utilizam um mecanismo de comunicação indireto, denominado feromônio, que induz as demais formigas a seguirem o caminho indicado. Esse comportamento emergente é resultado de um mecanismo de recrutamento, denominado coordenação indireta, onde formigas influenciam as demais para direções que possuem maior quantidade de feromônio.

Gambardella e Dorigo (1995) [7] desenvolveram o algoritmo *Ant-Q*, inspirados no algoritmo *Q-learning* desenvolvido por Watkins em 1992 [22]. No algoritmo *Ant-Q* o feromônio é denotado por *AQ-value* ($AQ(i,j)$). O objetivo do algoritmo *Ant-Q* é estimar $AQ(i,j)$ de tal forma que encontre soluções que favoreçam a coletividade. Os agentes selecionam suas ações baseados nas regras de transições, conforme as equações 1 e 2:

$$s = \begin{cases} \arg\max_{j \in N_k(t)} \left\{ [AQ(i,j)]^{\delta} \times [HE(i,j)]^{\beta} \right\} & \text{if } q \le q_0 \\ S & \text{caso contrário} \end{cases} \quad (1)$$

onde os parâmetros δ e β representam o peso (influência) do $AQ(i,j)$ e da heurística respectivamente; q é um valor selecionado aleatoriamente com distribuição de probabilidade [0,1], ou seja, quanto maior o valor de q_0 menor a probabilidade de escolher um estado aleatoriamente; S é uma variável aleatória selecionada de acordo com a probabilidade dada pela função $AQ(i,j)$; e $HE(i,j)$ são os valores heurísticos associados a ligação (aresta) (i,j), a qual auxilia na seleção dos estados (vértices) adjacentes (no caso do caixeiro viajante, é usado o inverso da distância euclidiana [7]).

Três diferentes regras foram propostas para selecionar o valor da variável aleatória S: i) *pseudo-aleatório*, onde S é um estado selecionado aleatoriamente do conjunto $N_k(t)$, de acordo com a distribuição uniforme; ii) *pseudo-aleatório-proporcional*, na qual S é selecionado de acordo com a distribuição apresentada na Equação 2 e; iii) *aleatório-proporcional*, dado que,

se q tenha assumido valor 0 na Equação 1, então o próximo estado será selecionado aleatoriamente baseado na distribuição conforme a Equação 2.

$$S = \begin{cases} \dfrac{[AQ(i,j)]^{\delta} \times [HE(i,j)]^{\beta}}{\sum_{a \in N_k(t)} [AQ(i,a)]^{\delta} \times [HE(i,a)]^{\beta}} \end{cases} \quad (\mathbf{2})$$

Gambardella e Dorigo (1995) [7] mostraram que uma boa regra para a escolha das ações com o algoritmo *Ant-Q* é baseado na *pseudo-aleatório-proporcional*. Dessa forma, o $AQ(i,j)$ é estimado ao utilizar a regra de atualização da Equação 3, similar ao algoritmo *Q-learning*:

$$AQ(i,j) \leftarrow (1-\alpha) \times AQ(i,j) + \alpha\left(\Delta AQ(i,j) + \gamma \times \max_{a \in S(j)} AQ(j,a)\right) \quad (\mathbf{3})$$

onde γ representa o fator de desconto e α a taxa de aprendizagem.

Quando a regra de atualização é local, a atualização do $AQ(i,j)$ é aplicada após o estado s ter sido selecionado, atribuindo valor 0 para $\Delta AQ(i,j)$. O efeito é que o $AQ(i,j)$ associado à ligação (i,j) é reduzido pelo fator γ cada vez que a ligação estiver na solução candidata. Assim como no *Q-learning* esta abordagem tende a desconsiderar a exploração de estados com menor probabilidade (feromônio), tornando o algoritmo inadequado para situações onde a solução atual deve ser alterada significativamente devido a alterações inesperadas do ambiente.

Outras técnicas baseadas em algoritmos de colônia de formigas foram propostas para melhorar a habilidade de exploração dos algoritmos em ambientes dinâmicos. Guntsch e Middendorf (2003) [12] propuseram uma técnica para melhorar a solução quando ocorrem alterações no ambiente, aplicando procedimentos de busca local para as novas soluções. Alternativamente, estados alterados são retirados da solução, conectando o estado predecessor e sucessor do estado excluído. Dessa forma, novos estados são inseridos na solução. O novo estado é inserido na posição que causa o custo mínimo ou diminui o custo mais alto (dependendo do objetivo) no ambiente. Sim e Sun (2002) [19] usaram múltiplas colônias de formigas, onde uma colônia é repelida pelo feromônio de outras colônias, favorecendo a exploração quando o ambiente é alterado. Outras técnicas para tratar a dinâmica no ambiente alteram a regra de atualização do feromônio para favorecer a exploração. Por exemplo, Li e Gong (2003) [15] modificaram as regras de atualização local e global do algoritmo *Ant Colony System*. A regra de atualização local foi alterada conforme a Equação 4.

$$\tau_{ij}(t+1) = (1 - p(\tau_{ij}(t)))\tau_{ij}(t) + \Delta\tau_{ij}(t) \quad (\mathbf{4})$$

onde $p_1(\tau_{ij})$ é uma função de τ_{ij} no tempo t, onde $\theta > 0$, *e.g.*:

$$p_1(\tau_{iJ}) = \frac{1}{1 + e^{-(\tau_{iJ} + \theta)}} \quad (5)$$

onde $\theta > 0$.

Tais técnicas podem ser usadas como alternativas para encontrar soluções quando o ambiente é modificado. Ao empregar regras de transições probabilísticas, o algoritmo de colônia de formigas aumenta a exploração do espaço de estados. Dessa forma, é utilizada a decisão de transição aleatória e alguns parâmetros são modificados, onde a nova informação heurística influencia a seleção das ligações mais desejáveis.

A dinâmica da evaporação faz com que valores elevados de feromônio sejam diminuídos. Assim, quando o ambiente é alterado e a solução não é a melhor, a concentração de feromônio nas ligações correspondentes diminui ao longo do tempo. A atualização global é realizada similarmente, mas considera somente a melhor e a pior solução global, *i.e.*:

$$\tau_{ij}(t+1) = (1 - \rho_2(\tau_{ij}(t)))\tau_{ij}(t) + \gamma_{ij}\Delta\tau_{ij}(t) \quad (6)$$

onde:

$$\gamma_{i,j} = \begin{cases} +1 & se\,(i,j)\,é\,a\,melhor\,solução\,global \\ -1 & se\,(i,j)\,é\,a\,pior\,solução\,global \\ 0 & caso\,contrário \end{cases} \quad (7)$$

Uma regra similar de atualização global também foi usada por Lee *et al*. 2001. Outras estratégias para modificar o valor do feromônio foram propostas para compensar a estagnação de algoritmos de colônia de formigas. Gambardella *et al*. [9] propuseram uma abordagem para reajustar os valores do feromônio com os valores distribuídos inicialmente. Em outra estratégia, Stutzle e Hoos [20] sugeriram aumentar proporcionalmente o valor do feromônio de acordo com a diferença em relação ao maior valor de feromônio.

Observou-se que várias técnicas baseadas em algoritmos de colônia de formigas foram desenvolvidas para melhorar a habilidade de exploração dos algoritmos em ambientes dinâmicos. Essas técnicas podem ser usadas como alternativas para melhorar a solução assim que o ambiente é alterado. As abordagens propostas são baseadas em técnicas que usam estratégias para melhorar a exploração usando a transição probabilística do *Ant Colony System*, aumentando a exploração do espaço de estados. Dessa forma, a decisão de transição mais aleatória é usada, variando alguns parâmetros onde a nova informação heurística influencia a seleção das arestas mais desejáveis.

Alguns trabalhos utilizam regras de atualização nas arestas da solução, incluindo um componente de evaporação similar à regra de atualização do *Ant Colony System*. Dessa forma, ao longo do tempo a concentração do feromônio diminui, fazendo que os estados menos favoráveis sejam menos explorados nos episódios futuros. Para isso, uma alternativa seria

reinicializar o valor do feromônio após observar as alterações no ambiente, mantendo uma referência para as melhores soluções encontradas. Se identificado o local da alteração no ambiente, o feromônio dos estados adjacentes é reinicializado, fazendo com que os estados se tornem mais desejados. Se um estado não é satisfatório, reforços podem ser menores (geralmente proporcional a qualidade da solução), e ao longo do tempo, tornam-se menos desejáveis devido à redução do feromônio pela evaporação.

É possível observar que a maioria dos trabalhos propostos concentra seus esforços em melhorar as regras de transição empregando estratégias sofisticadas para a convergência. No entanto, os experimentos mostram que tais métodos não conseguem bons resultados em ambientes altamente dinâmicos e onde o tamanho do espaço de busca é incerto. Na seção 3 são apresentadas as estratégias que foram desenvolvidas para a atualização de políticas geradas por recompensas (feromônios) para ambientes dinâmicos.

3 Estratégias para Atualização de Políticas para Algoritmos baseados em Colônias de Formigas

Apresentamos em [16] que algoritmos de colônia de formigas são eficientes quando os parâmetros de aprendizagem são ajustados adequadamente e quando não ocorrem modificações no ambiente que alteram a política ótima. No entanto, em ambientes dinâmicos não se tem a garantia da convergência do algoritmo *Ant-Q*. Antes de detalharmos as estratégias, nós sumarizamos o domínio da aplicação. É utilizado um problema de otimização combinatória frequentemente empregada na computação para demonstrar problemas de difícil resolução, o problema do caixeiro viajante. De uma maneira formal, o problema do caixeiro viajante é definido como um grafo (ambiente) completo $A=(E,L)$ com n estados (vértices) $E=\{e_1,...,e_n\}$, no qual L é o conjunto de todas as ligações (arestas) entre quaisquer dois estados i e j, sendo $l_{ij} = l_{ji}$ quando simétrico. O objetivo é encontrar o caminho de menor custo (distância), passando uma única vez por todos os estados sem repetição, até retornar ao estado de origem [10].

Uma alternativa para solucionar o problema do caixeiro viajante é testar as permutações possíveis, empregando algoritmos de busca exaustiva para encontrar o percurso com menor custo. No entanto, dado que a quantidade de permutações é $(n - 1)!$, tal alternativa torna-se impraticável para a solução na maioria das vezes. Portanto, diferentemente das técnicas exaustivas, algoritmos heurísticos como o *Ant-Q*, buscam soluções desejáveis em menor tempo. Mesmo sem garantir a melhor solução (política ótima), o ganho computacional favorece a solução.

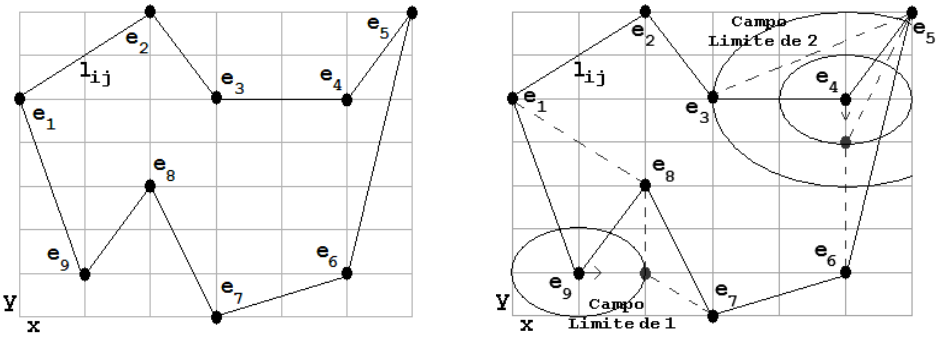
A convergência de algoritmos de colônia de formigas pode ocorrer se houver uma exploração exaustiva do espaço de estados, tornando o processo de aprendizagem bastante longo até a convergência. Além disso, agentes em ambientes dinâmicos podem ter políticas que atrasam o processo de aprendizagem ou que geram políticas subótimas. Apesar disso, a aceleração da convergência de algoritmos baseados em exames pode ser acelerada através do uso de políticas adaptativas, evitando atualizações não satisfatórias. Para isso, estratégias para

estimar a política corrente são apresentadas nos próximos parágrafos, melhorando a convergência dos agentes quando o ambiente é alterado.

As estratégias desenvolvidas modificam valores de feromônio, melhorando a coordenação entre os agentes e permitindo convergência mesmo quando há mudanças na posição cartesiana dos estados do ambiente. O objetivo das estratégias é encontrar o equilíbrio ótimo da recomposição da política, que permita explorar novas soluções usando informações de políticas passadas. Equilibrar o valor do feromônio equivale a reajustar as informações das ligações, dando ao processo de busca flexibilidade para encontrar uma nova solução quando o ambiente é alterado, compensando a influência das políticas passadas na construção de novas soluções.

Uma das estratégias de atualização desenvolvidas é inspirada nas abordagens propostas em [11] e [14], reinicializando localmente os valores de feromônio quando alterações no ambiente são identificadas. Este método é chamado de estratégia *média global*. Essa estratégia atribui às ligações adjacentes dos estados alterados, a média de todos os valores de feromônio da melhor política. A estratégia *média global* é limitada, pois não observa a intensidade de alteração do ambiente. Por exemplo: muitas vezes, boas soluções com estados alterados podem diminuir a qualidade da solução, sendo necessário atualizar apenas parte da política de ação. A estratégia *distância global* atualiza o feromônio dos estados considerando a distância euclidiana entre todos os estados do ambiente com a distância euclidiana do ambiente alterado. Se o custo da política aumenta com a alteração, então o valor do feromônio diminui proporcionalmente, caso contrário, o valor é aumentado. A estratégia *distância local* é similar à estratégia *distância global*, no entanto, a atualização do feromônio é proporcional à diferença na distância euclidiana dos estados que foram alterados.

Antes de discutirmos com mais detalhes como as estratégias atribuem valores para a política corrente, é apresentado como as alterações ocorrem no ambiente. Os estados do ambiente podem ser alterados devido a fatores como, escassez dos recursos, mudança de objetivos ou atribuições de tarefas, de tal maneira que estados podem ser inseridos, excluídos, ou simplesmente movimentados no ambiente. Tais características podem ser encontradas em diferentes aplicações como gerenciamento de tráfego, redes de sensores, gerenciamento de cadeias de suprimentos ou redes de comunicação móveis.



(a) Posição dos estados antes da alteração (A) (b) Posição dos estados após alteração (A')

Figura 1. Dinâmica do Ambiente

A Figura 1 mostra uma representação simplificada de um ambiente com 9 estados. A Figura 1a ilustra o ambiente antes da alteração, enquanto a Figura 1b ilustra o ambiente depois das alterações. A configuração do ambiente é mostrada na Tabela 1:

Tabela 1. Estados antes e após as alterações

Antes das alterações (A)		**Após alterações (A')**	
estados	ligações	estados	ligações
$e_1(0,5)$	1→2, 1→9	$e_1(0,5)$	1→2, **1→8**
$e_2(2,7)$	2→3, 2→1	$e_2(2,7)$	2→3, 2→1
$e_3(3,5)$	3→2, 3→4	$e_3(3,5)$	**3→5**, 3→2
$e_4(5,5)$	4→3, 4→5	**e_4(5,4)**	**4→6**, 4→5
$e_5(6,7)$	5→4, 5→6	$e_5(6,7)$	5→4, 5→3
$e_6(5,1)$	6→5, 6→7	$e_6(5,1)$	**6→4**, 6→7
$e_7(3,0)$	7→6, 7→8	$e_7(3,0)$	**7→9**, 7→6
$e_8(2,3)$	8→9, 8→7	$e_8(2,3)$	8→9, **8→1**
$e_9(1,1)$	9→1, 9→8	**e_9(2,1)**	**9→7**, 9→8

Pode-se observar que a alteração dos estados e_4 e e_9 irá atribuir à política corrente seis novas ligações. As alterações no ambiente são realizadas de maneira aleatória, considerando alterações na posição cartesiana dos estados mas restringindo-se ao tamanho do campo limite, ou seja, adjacentes à uma posição cartesiana.

Portanto, uma mudança introduzida no ambiente pode modificar a localização (posição) de um estado e isso pode causar diferenças parciais entre a política corrente e a política ótima, causando temporariamente políticas indesejadas e erros. As estratégias devem

atualizar o valor do feromônio de cada ligação dos estados alterados, conforme as características de cada estratégia.

A. Estratégia *Média Global*

A estratégia *média global* não considera a intensidade da alteração no ambiente, no entanto consegue perceber os estados alterados. É atribuído às ligações incidentes aos estados alterados o valor médio do feromônio de todas as ligações da melhor política corrente (Q). Diferentemente de outros trabalhos, que reinicializam o feromônio sem considerar o valor aprendido, a estratégia *média global* reutiliza os valores de políticas passadas para estimar os valores de atualização. A Equação 8 mostra como são computados os valores desta estratégia:

$$média\_global = \frac{\sum_{l \in Q} AQ(l)}{n_l} \quad (8)$$

onde n_l é o número de ligações e $AQ(l)$ é o valor do feromônio das ligações l.

B. Estratégia *Distância Global*

Na estratégia *distância global* é calculada a distância entre todos os estados e o resultado é comparado com a distância dos estados do ambiente alterado. Assim, esta estratégia considera a intensidade total de alteração no ambiente. Se a distância entre os estados aumenta, então a atualização do valor do feromônio é inversamente proporcional em relação à distância. Caso o custo da distância entre os estados diminua, então o valor é aumentado na mesma proporção. A Equação 9 é usada para estimar o valor de atualização nas ligações dos estados do ambiente A'.

$$distância\_global = \frac{\sum_{i=1}^{n_e} \sum_{j=i+1}^{n_e} d_A(l_{ij})}{\sum_{i=1}^{n_e} \sum_{j=i+1}^{n_e} d_{A'}(l_{ij})} \times AQ(l_{ij}) \quad (9)$$

onde n_e é o número de estados, A' o ambiente após as alterações e d a distância euclidiana entre os estados.

C. Estratégia *Distância Local*

A estratégia *distância local* é similar à estratégia anterior, no entanto atualiza o feromônio somente nas ligações incidentes aos estados modificados. Dessa forma, cada ligação é atualizada proporcionalmente à distância dos estados adjacentes que modificaram, tornando a atualização dessa estratégia local, melhorando a convergência quando ocorrem

poucas alterações no ambiente. A Equação 10 é usada para computar o valor da atualização nas ligações:

$$distância\_local = \frac{d_A(l_{ij})}{d_{A'}(l_{ij})} \times AQ(l_{ij}) \quad \textbf{(10)}$$

onde e' são os estados alterados do ambiente A'.

Na próxima seção é apresentado o *framework* e o pseudocódigo com as estratégias supracitadas.

4 FANTS - *Framework for Ants*

O FANTS foi desenvolvido para simular o algoritmo *Ant-Q* com as estratégias apresentadas na Seção 3. A Figura 2 apresenta uma visão geral do FANTS e seus principais componentes. Na Figura 2 é possível observar um ambiente em forma de *grid* que representa o local onde os estados são posicionados. A linha escura com maior intensidade representa a melhor política (melhor percurso) descoberta pelo algoritmo *Ant-Q*. Logo abaixo do *grid*, é apresentado o gráfico que mostra a evolução da aprendizagem do algoritmo, onde é possível observar o comportamento dos agentes ao longo dos episódios. Um episódio corresponde à uma sequência de ações, que determina os estados visitados pelos agentes, sendo finalizado quando os agentes retornam ao estado de origem após visitar todos os demais estados. A dimensão Y do gráfico é o custo do percurso de cada episódio. Já a dimensão X corresponde à quantidade de episódios. A linha do gráfico que possui maiores variações representa o percurso de menor custo de cada episódio, enquanto a segunda linha mostra o percurso de menor custo global de todos os episódios.

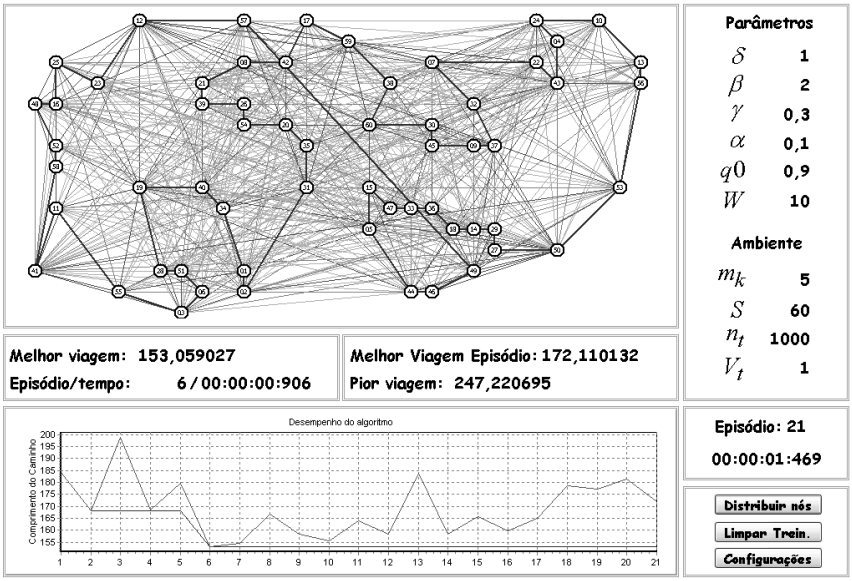


Figura 2. FANTS

À direita, na Figura 2, estão posicionados os parâmetros do algoritmo e do ambiente, onde δ e β são respectivamente os parâmetros da regra de transição e γ e α são parâmetros de aprendizagem do algoritmo. As variáveis m_k, S e n_t são o número de agentes, a quantidade de estados e o número de episódios respectivamente. O parâmetro n_t é usado como critério de parada do algoritmo. As estruturas internas do *framework* são formalizadas com as equações 8-13 que compõem o algoritmo *Ant-Q* apresentado no Pseudocódigo 1.

Inicialmente, é calculado com a Equação 11 o valor inicial para o feromônio:

$$\frac{1}{avg \times n} \tag{11}$$

$$d_{(ij)} = \sqrt{\left(i_x - j_x\right)^2 + \left(i_y - j_y\right)^2} \tag{12}$$

onde *avg* é a média das distâncias euclidianas dos estados pares (*i*,*j*) calculada pela Equação 12, e *n* é o número de agentes no sistema. Depois de calcular o valor do feromônio inicial, tal valor é atribuído às ligações que compõem os estados do grafo. Esse procedimento é realizado uma única vez, antes do primeiro episódio. Com isso, os agentes podem selecionar os estados baseados no valor do feromônio ou da heurística.

```
Algoritmo Ant-Q()
01   Início
02     Distribua os estados
03     Calcule o feromônio inicial com a equação 11 e o distribua
       nas ligações
04     Para cada episódio Repita:
05       Defina a posição inicial dos Agentes
06       Enquanto existirem estados a serem visitados Faça:
           // Nesse caso, lista tabu <> φ
07         Para cada Agente repita:
08           Se (q(rand(0..1) <= q0) Então
09             Escolha a ação conforme equação 1
10           Senão
11             Escolha a ação conforme equação 2
12           Fimse
13           Atualize o feromônio da ligação (i,j) usando a
             atualização local
14         Fimpara
15       Fimenquanto
16       Calcule o custo da melhor política do episódio
17       Realize a atualização global, usando as equações 3 e 13
18       Se ocorrer alterações no ambiente Então
19         Caso: (Estratégia média_global) Então
20           valor ← estrategia1(); // equação 8
21         Caso: (Estratégia distância_global) Então
22           valor ← estrategia2(); // equação 9
23         Caso: (Estratégia distância_local) Então
24           valor ← estrategia3(); // equação 10
25         Para cada estado alterado Faça:
26           Para cada ligação (i,j) incidente ao estado alte-
             rado Faça:
27             AQ(i,j) ← valor;
28           Fimpara
29         Fimpara
30       Fimse
31     Fimpara
32   Fim
```

Pseudocódigo 1. Algoritmo do FANTS com as estratégias

Um importante aspecto do algoritmo é a forma de atualização na tabela de aprendizagem, podendo ocorrer de maneira global ou local. A atualização global ocorre no final de cada episódio, onde é escolhida a política de menor custo e atualizados os valores dos estados com o parâmetro de reforço. A Equação 13 é usada para calcular o valor de $\Delta AQ(i,j)$, que será o reforço da atualização global.

$$\Delta AQ(i,j) = \begin{cases} \frac{W}{L_{best}} \end{cases} \quad (\mathbf{13})$$

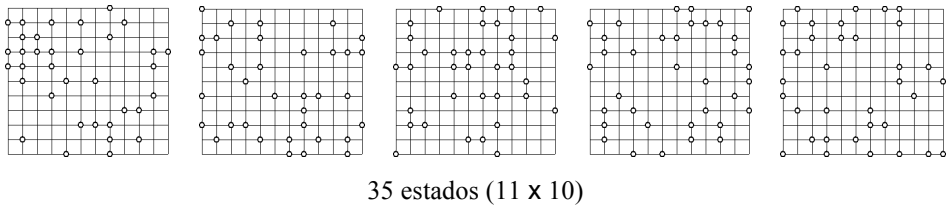
onde W é uma variável parametrizada com o valor 10 e L_{best} é o custo total do percurso. Já a atualização local ocorre a cada ação dentro do episódio, onde $\Delta AQ(i,j)$ terá valor zero.

5 Resultados Experimentais

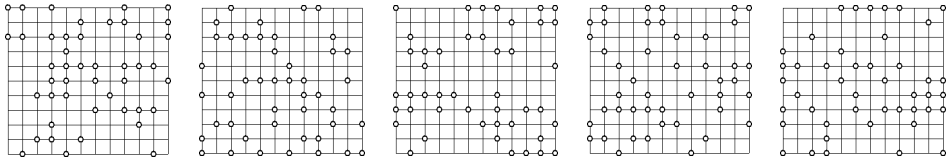
Experimentos são mostrados para avaliar as estratégias discutidas na Seção 3 e o impacto dos parâmetros de aprendizagem do algoritmo *Ant-Q*. Os experimentos realizados com o algoritmo avaliam sua eficiência ao considerar fatores como: (i) variações na taxa de aprendizagem; (ii) fator de desconto; (iii) taxa de exploração; (iv) regras de transição; (v) quantidade de agentes no sistema; e (vi) as estratégias propostas. As subseções 5.1 à 5.6 apresentam os resultados.

Para analisar a eficiência dos parâmetros do algoritmo, foram gerados 5 cenários diferentes para cada tipo de experimento, em ambientes de 35, 45 e 55 estados (Figura 3). O aprendizado em cada cenário foi realizado 15 vezes pelo algoritmo (15 amostras), pois se observa que fazendo experimentos em um mesmo ambiente, com entradas iguais, podem ocorrer variações na eficiência gerada pelo algoritmo. Isto ocorre porque as ações dos agentes são probabilísticas e os valores gerados durante sua aprendizagem são estocásticos. Portanto, as políticas de ação dos agentes podem variar de um experimento para outro. Assim, a eficiência apresentada nesta seção representa a média de todos os experimentos gerados nos 5 cenários com 15 amostras em cada ambiente. Esse número de repetições foi suficiente para avaliar a eficiência do algoritmo, pois observamos que a partir deste número os resultados dos experimentos não alteravam significativamente a qualidade das políticas. O eixo Y dos gráficos das figuras 5 a 9 apresenta o custo da política em % encontrada com cada parâmetro. O eixo X do gráfico indica o valor do parâmetro. Em muitos problemas de otimização não é possível *a priori* conhecer a política ótima. Para calcular a eficiência do algoritmo em percentual (eixo Y), é usada uma escala para cada política, onde 100% indica a política de menor custo e 0 caso contrário.

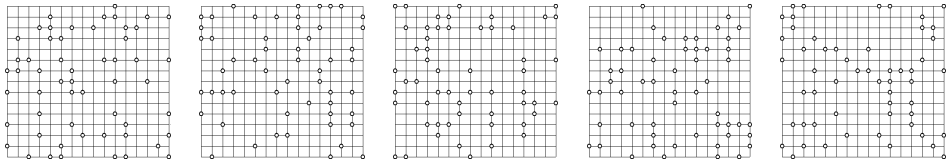
A quantidade de agentes no ambiente é igual à quantidade de estados. Inicialmente, os parâmetros foram configurados com os seguintes valores: δ= 1; β= 2; γ= 0,3; α= 0,1; q_0= 0,9 e W= 10. Foi utilizada como critério de parada a quantidade de 300 episódios. Cabe observar que dependendo do tamanho e da complexidade do ambiente, esse número não é suficiente para encontrar a melhor política. No entanto, o objetivo dos experimentos é avaliar o impacto dos parâmetros na convergência dos agentes e não a qualidade da política encontrada.



35 estados (11 x 10)



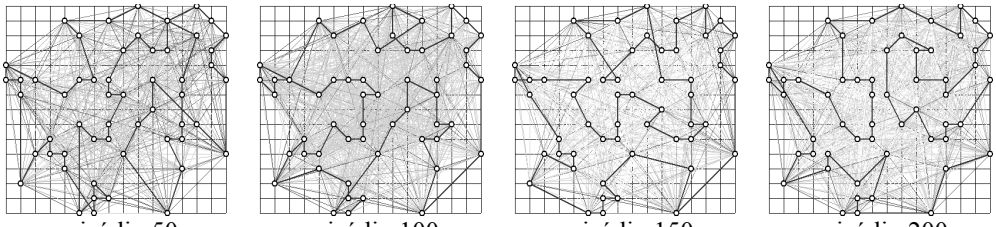
45 estados (11 x 10)



55 estados (15 x 14)

Figura 3. Ambientes usados na simulação, onde os estados estão expressos em um sistema euclidiano de coordenadas 2D

Antes de discutirmos os resultados variando os fatores de aprendizagem, é importante lembrar que observamos a necessidade de poucos episódios para encontrar as melhores políticas. Isso acontece devido à influência da heurística, que foi parametrizada com o dobro do valor da influência do feromônio. As imagens da Figura 4 ilustram a evolução da política em um ambiente com 55 estados a cada 50 episódios.



episódio 50 episódio 100 episódio 150 episódio 200

Figura 4. Evolução da política a cada 50 episódios

Os resultados apresentados nas subseções 5.1 à 5.5 correspondem à versão original do algoritmo *Ant-Q*, enquanto os experimentos sobre ambientes dinâmicos e as estratégias de atualização são apresentados a partir da subseção 5.6.

5.1. Taxa de Aprendizagem

A taxa de aprendizagem α indica a importância do feromônio computado ao estado selecionado. Para verificar os melhores valores para α foram realizados experimentos nos três ambientes usando valores entre 0 e 1. Os melhores valores para α estão entre 0,1 e 0,2. Valores superiores fazem com que, os agentes ao estabelecerem uma melhor ação em um determinado estado do ambiente, não efetuassem outras ações na busca de caminhos de menor custo. Valores inferiores não dão a devida importância ao aprendizado, não permitindo que os agentes selecionem caminhos diferentes da política corrente. O melhor valor de α para a política foi de 0,1, sendo usado nos demais experimentos. Observamos ainda que, quanto menor a taxa de aprendizagem, menor é a variação da política. A Figura 5 apresenta a eficiência das taxas de aprendizagem no intervalo [0,.,1].

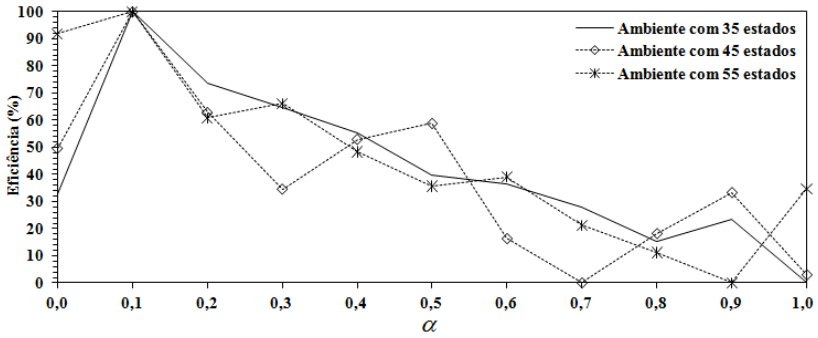


Figura 5. Eficiência da taxa de aprendizagem (α)

5.2. Fator de Desconto

O fator de desconto determina o peso temporal relativo dos reforços recebidos. Os melhores valores para o fator de desconto estão entre 0,2 e 0,3 conforme apresentado na Figura 6. Valores diferentes de 0,2 e 0,3 mostraram-se ineficientes para a convergência, tendo pouca relevância para a aprendizagem dos agentes. Quando o valor é superior a 0,3, ele apresenta relevância excessiva, induzindo os agentes a ótimos locais.

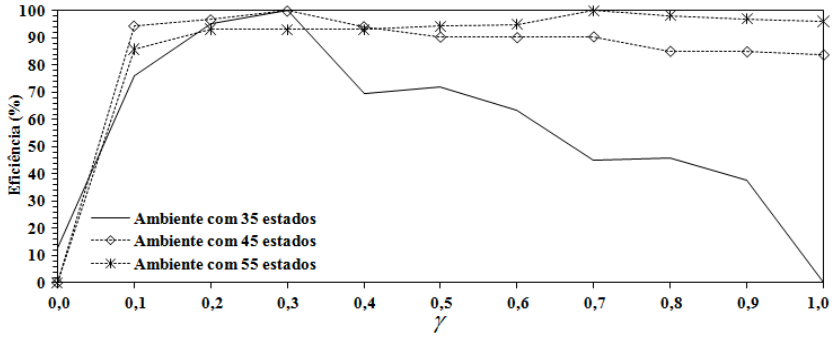


Figura 6. Eficiência do fator de desconto (γ)

5.3. Taxa de Exploração

A taxa de exploração q_0 indica a probabilidade de um agente escolher um determinado estado. Os experimentos para encontrar a melhor taxa de exploração foram realizados em ambientes com estados e tamanhos diferentes. Os melhores valores utilizados estão entre 0,8 e 1. À medida que o valor se aproxima de zero, as ações dos agentes vão se tornando cada vez mais aleatórias, consequentemente as soluções começam a não ser satisfatórias.

O melhor valor encontrado para q_0 é 0,9. Com isso, agentes selecionam os caminhos de menor custo e com maior concentração de feromônio. Com q_0= 0,9 a busca é praticamente gulosa, pois 0,1 será a probabilidade de escolher os demais caminhos. A Figura 7 mostra os resultados para q_0 no intervalo [0,.,1].

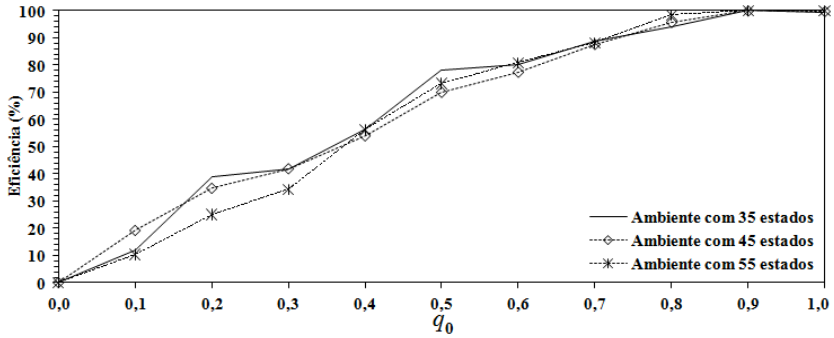


Figura 7. Resultados do parâmetro de exploração (q_0)

5.4. Regra de Transição

Os experimentos alterando os fatores δ e β foram realizados em ambientes com estados e tamanhos diferentes. Conforme observado, o algoritmo é dependente de heurísticas,

onde o peso é representado pelo parâmetro β. Para obter bons resultados, o valor de β deve ser pelo menos 65% do valor de δ. A Figura 8 ilustra os resultados variando os fatores δ e β.

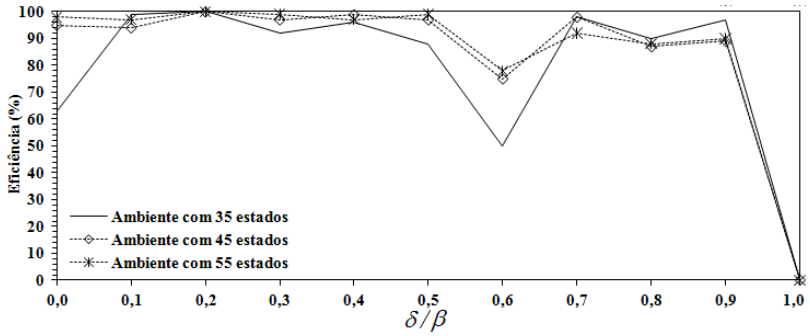


Figura 8. Resultados da regra de transição (δ e β)

5.5. Quantidade de Agentes

Para avaliar o impacto da quantidade de agentes no sistema, foram utilizados de 10 a 80 agentes. A Figura 9 mostra que as melhores políticas são encontradas quando a quantidade de estados é igual à quantidade de agentes no sistema ($m_k = x$), onde x é a quantidade de estados e x_i é a variação dos agentes no sistema. Pode-se observar que a quantidade superior de agentes ao número de estados ($m_k > x$) mostra-se inadequada para boas soluções, apresentando comportamento de estagnação. Assim, ao encontrarem uma solução, agentes evitam a busca por outros caminhos, determinando um máximo local. Quando a quantidade de agentes é inferior ao número de estados ($m_k < x$), o número de episódios teve que ser aumentado de maneira exponencial para encontrar as melhores soluções.

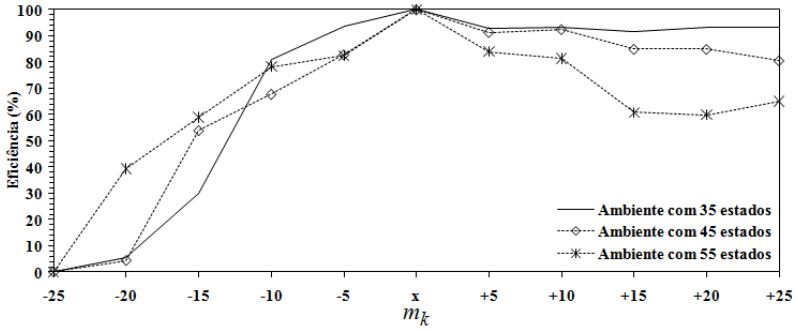


Figura 9. Quantidade de agentes (m_k)

5.6. Desempenho dos Agentes com as Estratégias de Atualização

Para avaliar as estratégias propostas na Seção 3, foram gerados ambientes dinâmicos com 35 estados. O comportamento dos agentes foi avaliado considerando a porcentagem de mudança gerada pelo ambiente a cada 100 episódios. Essa janela temporal foi utilizada porque em trabalhos anteriores foi observado que em ambientes de 35 estados ela permitiu ao algoritmo conquistar boa convergência [16].

A alteração ocorre da seguinte maneira: a cada 100 episódios, o ambiente produz um conjunto de alterações. As mudanças são realizadas aleatoriamente, de tal maneira que simule alterações em locais parcialmente conhecidos ou sujeitos a ruído. Dessa forma, ambientes com 35 estados terão 7 estados alterados quando 20% de mudança ocorrer. Ademais, foram simuladas alterações considerando o espaço do campo limite com profundidade 1 e 2, limitando assim a mudança da posição de um estado e permitindo simular dinâmicas graduais próximas de problemas do mundo real.

Os resultados dos experimentos comparam as três estratégias com a política descoberta com o *Ant-Q* original. Os parâmetros de aprendizagem utilizados na simulação são os melhores encontrados nas subseções 5.1 a 5.5. Cada estratégia permitiu que na maioria das vezes o número de episódios diminuísse, pois a combinação das recompensas pôde estimar valores melhores, que levaram os agentes a uma convergência quando a política é atualizada. As figuras 10, 11, 12 e 13 demonstram a convergência do algoritmo em ambientes com 35 estados. O eixo X dessas figuras indica os episódios. Quando o percurso de menor custo é encontrado, a eficiência é 100% (eixo Y).

Observando as figuras 10, 11, 12 e 13, é possível notar que a política global com as estratégias é superior a do *Ant-Q* original. A estratégia *média global* mostra-se mais adequada para ambientes com variações maiores (figuras 11 e 13). Isso ocorre porque a estratégia utiliza todos os valores de reforços do ambiente. No entanto, os agentes sofrem para convergir quando o ambiente tem poucas alterações, pois estados alterados terão recompensas menores que os estados que constituem a melhor solução atual. Já a estratégia *distância global* mostra-se mais robusta em ambientes com poucas variações (figuras 10 e 12). Quando o ambiente é alterado, a estratégia age nos estados atualizando a recompensa proporcionalmente à quantidade de alterações do ambiente. Dessa forma, o efeito da atualização diminui o impacto após as mudanças, fazendo que os agentes convergissem uniformemente. A estratégia *distância local* considera somente as alterações locais, dessa forma, a atualização da política com tal estratégia é melhor quando os valores dos reforços são maiores, ou seja, nos episódios finais.

De maneira geral, a política global das estratégias consegue acumular bons valores de reforços com um número pequeno de episódios de aprendizagem. As estratégias atualizam a política global acumulando bons valores de reforços, desde que haja uma quantidade de episódios necessária. Nos episódios iniciais da aprendizagem, a política é menos sensível às estratégias, o que melhora o desempenho da política após a atualização. Algumas estratégias

podem estimar valores não adequados para a política, principalmente após muitos episódios e mudanças no ambiente, ocasionando máximos locais.

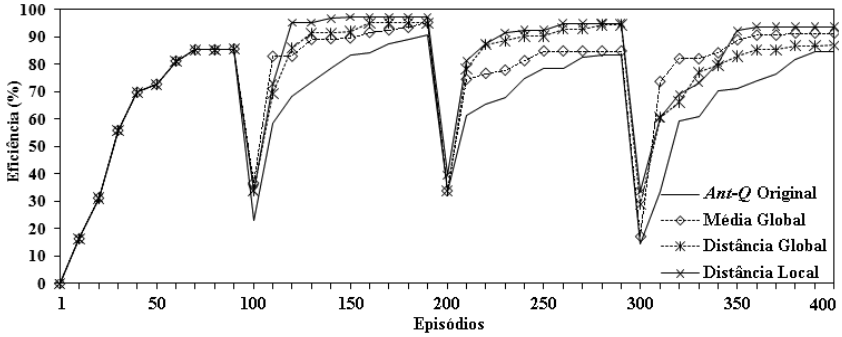


Figura 10. Campo limite de 1; 10% de alterações a cada 100 episódios

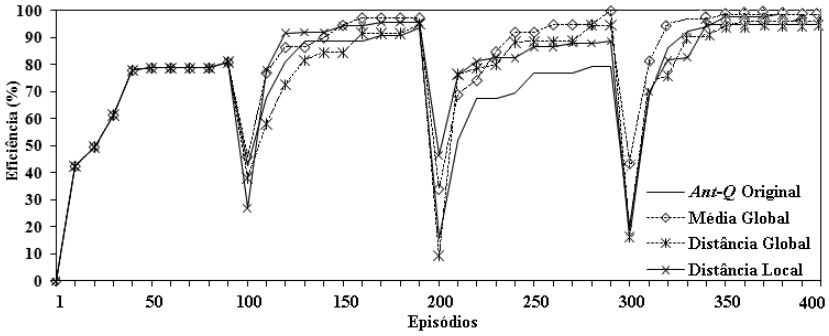


Figura 11. Campo limite de 1; 20% de alterações a cada 100 episódios

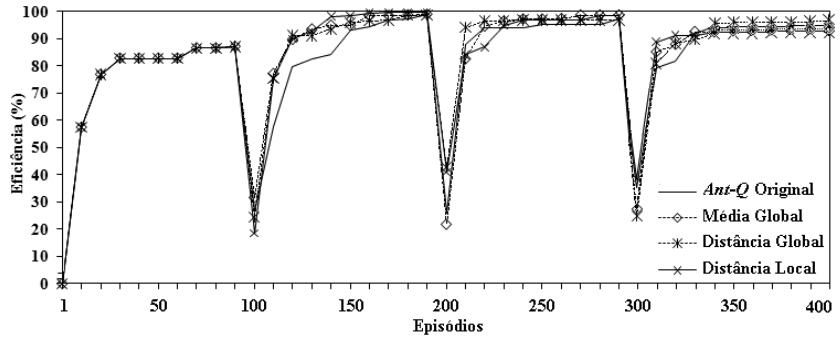


Figura 12. Campo limite de 2; 10% de alterações a cada 100 episódios

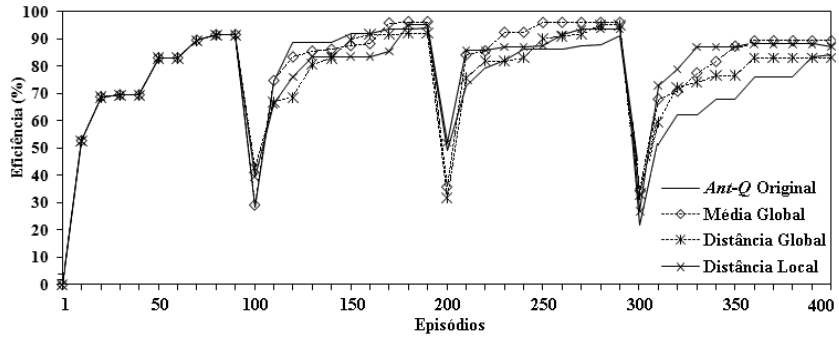


Figura 13. Campo limite de 2; 20% de alterações a cada 100 episódios

Uma observação interessante é o impacto do campo limite (adjacentes à posição cartesiana) nas estratégias. Mesmo com campo limite restrito, as estratégias melhoram a convergência do algoritmo. Em outros experimentos com o campo limite igual a 5, a eficiência do algoritmo *Ant-Q* é inferior (21%) quando comparado com a melhor estratégia (figuras 14 e 15).

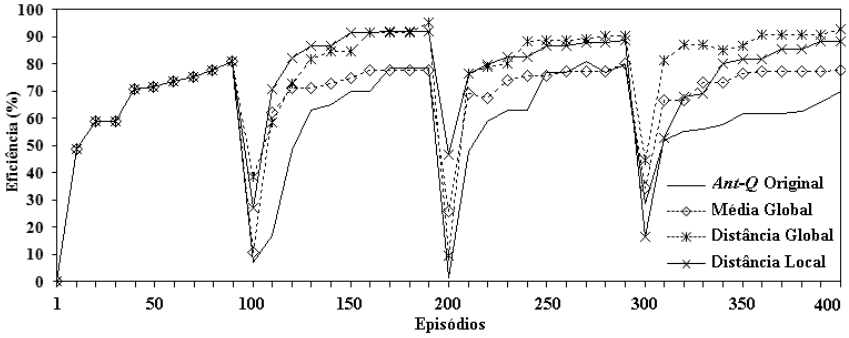


Figura 14. Campo limite de 5; 10% de alterações a cada 100 episódios

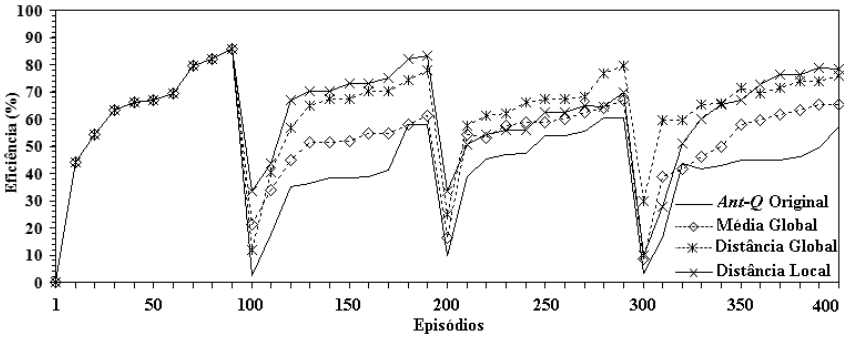


Figura 15. Campo limite de 5; 20% de alterações a cada 100 episódios

A estratégia *média global* é mais adequada quando o campo limite é inferior a 5 (figuras 10 a 13). Como a atualização é feita com a média de todos os valores de feromônio, o valor das ligações dos estados alterados é equalizado. As estratégias *distância global* e *distância local* podem convergir rapidamente quando o campo limite é igual 5 (figuras 14 e 15). Isso ocorre porque a atualização é proporcional à distância de cada ligação incidente nas ligações dos estados alterados. Assim, as ligações que não pertencem mais à política Q^*, terão o valor do feromônio enfraquecido.

6 Conclusões e Considerações Finais

Técnicas de coordenação derivadas da aprendizagem por reforço vêm sendo estudadas nesses últimos anos por diversos pesquisadores, que descrevem diferentes aplicações no uso de agentes inteligentes [17], [21] e [22]. Nesse paradigma, a aprendizagem ocorre quando um agente aprende por tentativa e erro ao interagir com o ambiente ou entre si. A fonte de aprendizado é a própria experiência do agente, cujo objetivo é adquirir uma política de ações que maximize seu desempenho geral.

A coordenação adequada dos agentes que empregam algoritmos de aprendizagem depende dos ajustes dos parâmetros para encontrar as melhores soluções. Dessa forma, observa-se que técnicas de otimização por enxames são baseadas em recompensas (feromônio) que influenciam o comportamento dos agentes, gerando políticas que melhoram a coordenação e o comportamento global do sistema.

A aplicação de agentes de aprendizagem no problema da coordenação de sistemas multiagente tem se tornado cada vez mais frequente. Isso ocorre porque a adaptação dos modelos de coordenação geralmente é necessária em problemas complexos, eliminando e/ou reduzindo deficiências dos mecanismos de coordenação tradicionais [6]. Dessa forma, foi apresentado neste artigo o FANTS, um *framework* de teste para analisar o desempenho dos agentes com o algoritmo *Ant-Q* e para descrever o comportamento do *Ant-Q* com diferentes

cenários, parâmetros e estratégias de atualização de políticas para ambientes dinâmicos (discutidas na Seção 3). O *framework* apresentado é capaz de mostrar de maneira interativa o impacto da variação dos parâmetros e da quantidade de agentes nas abordagens, possibilitando conhecer os valores adequados dos parâmetros do *Ant-Q*.

Os resultados obtidos a partir da utilização das estratégias de atualização de políticas para ambientes dinâmicos mostram que o desempenho do algoritmo *Ant-Q* é superior ao desempenho da política global descoberta sem as estratégias. Apesar das particularidades de cada estratégia, os agentes conseguem melhorar a política com atualizações globais e locais, mostrando que as estratégias podem ser usadas em ambientes dinâmicos.

Experimentos realizados com as estratégias mostram que mesmo tendo um custo computacional mais elevado, seus resultados são satisfatórios, pois as melhores soluções são encontradas em um número menor de episódios. Apesar dos resultados, experimentos adicionais são necessários para responder algumas questões em aberto. Por exemplo, a coordenação poderia ser realizada utilizando apenas os parâmetros mais significativos. Uma função heurística poderia ser usada para a aceleração do *Ant-Q*, onde indicando a escolha da ação tomada e limitando o espaço de busca do sistema. A atualização da política poderia ser feita por outras técnicas de coordenação, evitando a estagnação e máximos locais. Algumas dessas estratégias são encontradas em [17]. Outra questão consiste em avaliar o algoritmo em ambientes com maior número de estados. Essas hipóteses e direcionamentos deverão ser verificados em pesquisas futuras.

Agradecimentos

Nós agradecemos aos revisores anônimos por seus comentários úteis e aos membros do Laboratório de Agentes de Software da Pontifícia Universidade Católica do Paraná (PUCPR). Agradecimento em especial ao prof. Dr. Fabrício Enembreck pelos direcionamentos e comentários. Esta pesquisa é apoiada pela Financiadora de Estudos e Projetos (FINEP), sob o número 3560/06, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), CNPQ (Conselho Nacional de Desenvolvimento Científico e Tecnológico), processo 470325/2008-9 e em parte pela Universidade do Contestado (UnC) e Universidade Tecnológica Federal do Paraná (UTFPR).

Referências

[1] Costa, D. and Hertz, A. (1997) “Ants can colour graphs”, Journal of the Operational Research Society 48, p. 295-305.

[2] Culler, D., Estrin D., and Srivastava M. (2004) “Overview of sensor networks”. IEEE Computer, 37(8):41-19.

[3] Dorigo, M. (1992) “Optimization, Learning and Natural Algorithms”, PhD thesis, Politecnico di Milano, Itália.

[4] Dorigo, M. and Gambardella, L. M. (1996) "A Study of Some Properties of Ant-Q", In Proceedings of PPSN Fourth International Conference on Parallel Problem solving From Nature , p. 656-665.

[5] Dorigo, M., Maniezzo, V. and Colorni, A. (1996) "Ant System: Optimization by a Colony of Cooperting Agents", IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1):29-41.

[6] Enembreck, F.; Ávila, B. C.; Scalabrin, E. E. and Barthes, J. P. (2009) "Distributed Constraint Optimization for Scheduling in CSCWD". In: Int. Conf. on Computer Supported Cooperative Work in Design, 2009, Santiago, v. 1. p. 252-257.

[7] Gambardella, L. M. and Dorigo, M. (1995) "Ant-Q: A Reinforcement Learning Approach to the TSP", In proc. of ML-95, Twelfth Int. Conf. on Machine Learning, p. 252-260, 1995.

[8] Gambardella, L. M., Montemanni, R., Rizooli, A. and Donati, A. (2005) "Ant colony System for a Dynamic Vehicle Routing Problem", Journal of Combinatorial Optimization, v. 10(4), p. 327-343(17).

[9] Gambardella, L. M., Taillard, E. D. and Dorigo, M. (1997) "Ant Colonies for the QAP", Technical report, IDSIA, Lugano, Switzerland.

[10] Goldbard, M. and Luna, H. P. L. (2005) "Otimização Combinatorial e Programação linear: modelos e algoritmos", Rio de Janeiro: Campus.

[11] Guntsch, M. and Middendorf, M. (2001) "Pheromone Modification Strategies for Ant Algorithms Applied to Dynamic TSP", In Proc. of the Workshop on Applications of Evolutionary Computing, p. 213-222.

[12] Guntsch, M. and Middendorf, M. (2003) "Applying Population Based ACO to Dynamic Optimization Problems", In Proc. of Third Int. Workshop ANTS 2002, p. 111-122.

[13] Kennedy, J., Eberhart, R. C., with Shi, Y. (2001) "Swarm Intelligence". Morgan Kaufmann / Academic Press.

[14] Lee, S. G., Jung, T. U. and Chung, T. C. (2001) "Improved Ant Agents System by the Dynamic Parameter Decision", In Proc. of the IEEE Int. Conf. on Fuzzy Systems, p. 666-669.

[15] Li, Y. and Gong, S. (2003) "Dynamic Ant Colony Optimization for TSP", International Journal of Advanced Manufacturing Technology, 22(7-8):528-533.

[16] Ribeiro, R., Ronszcka, A. F., Borges, A. P., Enembreck, F. (2009) "Otimização dos Parâmetros de Aprendizagem para a Coordenação dos Agentes em Algoritmos de Enxames". In: Simpósio de Computação Aplicada (SCA 2009), 2009, Passo Fundo. Simpósio de Computação Aplicada (SCA 2009).

[17] Ribeiro, R., Borges, A. P. and Enembreck, F. (2008) "Interaction Models for Multiagent Reinforcement Learning", Int. Conf. on Computational Intelligence for Modelling Control and Automation - CIMCA08, Vienna, Austria, p. 1-6.

[18] Ribeiro, R., Enembreck, F. and Koerich, A. L. (2006) "Uma Nova Metodologia para Avaliação da Performance de Algoritmos Baseados em Aprendizagem por Reforço", XXXIII SEMISH, Campos Grande, MS, p. 433-446.

[19] Sim, K. M. and Sun, W. H. (2002) "Multiple Ant-Colony Optimization for Network Routing", In Proc. of the First Int. Symposium on Cyber Worlds, p. 277-281.

[20] Stutzle, T. and Hoos, H. (1997) "MAX-MIN Ant System and Local Search for The Traveling Salesman Problem", In Proceedings of the IEEE International Conference on Evolutionary Computation, 1997, p. 309-314.

[21] Tesauro, G. (1995) "Temporal difference learning and TD-Gammon", Communications of the ACM, vol. 38 (3), p. 58-68.

[22] Watkins, C. J. C. H., Dayan, P. (1992) "Q-Learning", Machine Learning, vol.8(3), p.279-292.

[23] Wooldridge, M. J. (2002) "An Introduction to MultiAgent Systems", John Wiley and Sons.

[24] Yuhui Shi, Russell C. (2009) "Eberhart: Monitoring of particle swarm optimization". Frontiers of Computer Science in China 3(1): 31-37.