ALGORITMOS DE OTIMIZAÇÃO BASEADOS EM ENXAMES INTELIGENTES[1]

Adriano F. Ronszcka[2]
Mário Gonçalves Júnior[3]
Richardson Ribeiro[4]

RESUMO: Este artigo apresenta um método de otimização para coordenar as ações dos agentes no Problema do Caixeiro Viajante em Sistemas Multi-Agente. A coordenação do Sistema Multi-Agente é necessária quando os recursos são limitados e as informações compartilhadas são essenciais para a cooperação entre grupos de agentes. Distribuir recursos e coordenar as ações dos agentes em ambientes do mundo real é uma tarefa complexa, devido à dinâmica e as características dos agentes. Neste trabalho foi utilizado um algoritmo baseado em Enxames (Colônia de Formigas) capaz de acelerar a convergência dos agentes no sistema. O método foi testado em ambientes dinâmicos e estocásticos, permitindo a avaliação do impacto dos aspectos que afetam a performance da abordagem proposta, como a quantidade de agentes no ambiente,os parâmetros de aprendizagem do algoritmo, entre outros. Resultados experimentais mostram a generalidade e a robustez do algoritmo.

Palavras-chave: Colônia de Formigas, Métodos de Coordenação, Sistemas Multi-agente.

ABSTRACT: This paper presents an optimization method to coordinate agents' actions in the Travelling Salesman Problem in Multi-Agent Systems. The coordination of Multi-Agent System is necessary when resources are limited and the information shared is essential to cooperation among groups of agents. Distributing resources and coordinating the agents' actions in real world environments is a complex task due to the dynamics and characteristics of agents. In this paper we have used Swarm Intelligence algorithm (Ant Colony) able to accelerate the convergence of agents into system. The method has been tested in dynamic and stochastic environments, allowing the evaluation of the impact of the aspects that affect the performance of the proposed approach, such as the number of agents in the environment, the learning parameters of the algorithm, and so on. The experimental results have show the generality and robust of algorithm.

Key Words: Ant Colony, Coordination methods, Multi-Agent Systems.

INTRODUÇÃO

A coordenação das ações dos agentes, quando bem aplicada, pode contribuir para a execução eficiente das tarefas realizadas por Sistemas Multi-Agente (SMA). A coordenação das ações dos agentes pode ajudar a evitar problemas como: soluções redundantes no mesmo subproblema, inconsistência de execução (atuação sobre subproblemas obsoletos); desperdício de recursos e "*deadlocks*" (espera) por eventos que provavelmente não irão ocorrer (WOOLDRIDGE, 2002).

A coordenação efetiva é essencial para que agentes alcancem seus objetivos. Para isso, métodos de coordenação devem ser utilizados para gerenciar as diversas formas de dependência que ocorrem quando agentes têm objetivos inter-relacionados, quando compartilham um ambiente em comum ou quando compartilham recursos.

Para produzir as melhores soluções a partir dos recursos e informações disponíveis, uma variedade de técnicas de coordenação para SMA foram propostas, por exemplo: leis sociais (OSSOWSKI, 1999); estrutura organizacional (BOURON, 1992); protocolos de mercado, como *contract-net* (SMITH e DAVIS, 1983); coordenação reativa (BARAY, 1999); formação de coalizão (KETCHPEL, 1993); planejamento distribuído (CORKILL, 1979) como *Partial Global Planning* (*PGP*) (DURFEE, 1999); algoritmos de otimização distribuída de restrições (*DCOP*) (LESSER *et al*., 2003), entre outros.

Um novo paradigma de coordenação, baseado em Enxames Inteligentes (*Swarm Intelligence*), tem sido estudo nessa última década. O paradigma é inspirado em Colônias de Insetos Sociais, que consiste de sistemas com comportamento similar ao de formigas, abelhas, cupins ou vespas. Tais colônias possuem características desejáveis para a solução de diversos problemas computacionais, que precisam de coordenação. Pesquisas anteriores sobre a organização de Colônias de Insetos Sociais e suas aplicações na organização de SMA mostram bons resultados em problemas complexos de otimização combinatória (DORIGO *et al.*, 1996).

A aplicação de aprendizagem para o problema de coordenação em SMA tem se tornado popular na Inteligência Artificial Distribuída (IAD). O uso de sistemas baseados em insetos, em especial algoritmos de Colônia de Formigas (CF), - *ant colony*, tem atraído a atenção de pesquisadores, por ser um meio atrativo para conquistar comportamentos coordenados por causa de sua generalidade e robustez (GAMBARDELLA e DORIGO, 1996).

Algoritmos de CF, como *Ant System* (DORIGO, 1992) e *Ant Colony System* (GAMBARDELLA e DORIGO, 1996) têm sido aplicados diretamente com algum sucesso em problemas como: caixeiro viajante (DORIGO, 1992), coloração de grafos (COSTA e HERTZ, 1997), roteamento de veículos (GAMBARDELLA *et al.*, 2005), entre outros.

OTIMIZAÇÃO POR COLÔNIA DE FORMIGAS

A Otimização por Colônia de Formigas (*Ant Colony Optimization - ACO*) (DORIGO, 1992) é uma abordagem bem sucedida baseada em população que tem sido aplicada em problemas de otimização combinatória NP-*hard* (BULLNHEIMER, 1999). Em outras palavras, a ACO é uma metaheurísti-

ca[1] para a solução de problemas combinatórios, inspirada no comportamento forrageiro de um grupo de formigas (agentes) na busca por alimentos. Para problemas de otimização combinatória é difícil encontrar uma solução ótima, e à medida que ocorre o aumento de entrada de dados o custo computacional cresce exponencialmente (DORIGO *et al.*, 1996).

Estudos sobre o comportamento forrageiro entre várias espécies de formigas revelaram que elas seguem um padrão de decisão baseada na aleatoriedade (DORIGO *et al.*, 1996). À medida que uma fonte de alimento é localizada, as formigas utilizam um mecanismo de comunicação indireto, denominado feromônio, que induzem as demais a seguirem o mesmo caminho. Este comportamento emergente é resultado de um mecanismo de recrutamento, onde formigas influenciam outras a seguirem em direção as fontes de alimento pelo caminho mais curto (GAMBARDELLA *et al.*, 1997).

Quando uma formiga localiza uma fonte de alimento, ela carrega a comida até o ninho e vai depositando feromônio ao longo do caminho. As formigas optam em seguir diferentes caminhos baseado nas concentrações de feromônio encontradas no ambiente. Desta forma, quanto maior a concentração de feromônio, maior a probabilidade de o caminho ser escolhido. Dessa forma, quanto mais formigas optam por seguir um caminho específico, mais reforçada é a qualidade do mesmo, atraindo assim mais formigas nesse caminho.

A figura 1 ilustra uma experiência feita por Goss *et al.* (1989) para estudar o comportamento das formigas. Inicialmente, elas exploram aleatoriamente a área ao redor do formigueiro à procura de comida. Enquanto se deslocam, depositam no ambiente uma quantidade de feromônio, indicando o caminho de volta ao formigueiro. Desta forma, quando uma formiga estabelece uma trilha ou caminho da fonte de alimento ao formigueiro, o caminho percorrido é identificado pelo feromônio. Assim, as demais formigas podem detectar a presença do feromônio no ambiente, escolhendo os caminhos com a maior concentração.

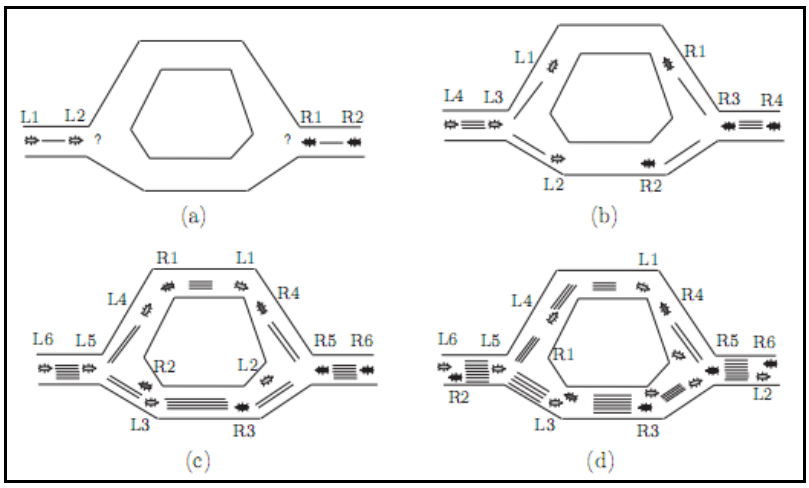

Figura 1 - Comportamento de formigas reais
Fonte: (GOSS *et al.*, 1989)

Na figura 1(a) as formigas localizam no ambiente um local com alternativas diferentes para alcançar o ninho.. Na figura 1(b) as formigas escolhem de maneira aleatória o caminho a seguir. As formigas que escolheram o caminho mais curto (de menor custo) alcançam o objetivo mais rapidamente (figura 1(c)). Já na figura 1(d) o caminho mais curto apresenta maior concentração de feromônio (representado pela quantidade de linhas no caminho das formigas).

[1] As metaheurísticas são procedimentos destinados a encontrar uma boa solução, eventualmente a ótima, consistindo na aplicação, em cada passo, de uma heurística subordinada, a qual tem que ser modelada para cada problema específico.

Portanto, as formigas que escolhem o caminho mais curto farão o percurso em menor tempo e a concentração de feromônio será reforçado com maior freqüência que nos caminhos mais longos. Assim, os caminhos mais eficientes, ou seja, de menor distância, receberão maior quantidade de feromônio e tendem a serem os mais escolhidos. Por ser uma substância volátil, a evaporação do feromônio evita que, com o tempo, um caminho que não esteja sendo mais utilizado continue a influenciar a decisão das formigas.

Estudos têm sido realizados no intuito de obter um melhor entendimento de como tais indivíduos tem sucesso exibindo um comportamento emergente complexo. O primeiro algoritmo inspirado em CF tem origem no trabalho de Dorigo *et al*. (1992) que propôs um algoritmo chamado de *Ant System* para solucionar o Problema do Caixeiro Viajante, e desde então vários aprimoramentos têm sido desenvolvidos para tal problema. Dentre esses podemos citar o *Ant-Q* (DORIGO e GAMBARDELLA, 1996) e (GAMBARDELLA e DORIGO, 1995) e *Ant Colony System* (BONABEAU *et al*., 1999), *Max-Min Ant System* (STÜTZLE e HOOS, 1997), *Antabu* (ROUX *et al*., 1998) e uma série de variantes desses algoritmos.

PROCEDIMENTOS METODOLÓGICOS

Antes de apresentarmos o framework, nós sumarizamos o Problema do Caixeiro Viajante (PCV). O PCV é um problema de otimização combinatória, frequentemente utilizado na computação para demonstrar problemas de difícil resolução (GOLDBARG e LUNA, 2005).

De uma maneira mais formal, o PCV simétrico é definido como um grafo completo $G = (V, E)$ com *n* vértices $V = \{v_1, ..., v_n\}$, no qual *E* é o conjunto de todos os arcos de *i* até *j*, sendo $d_{ij} = d_{ji}$. O objetivo é encontrar o caminho de menor distância, visitando todas as cidades sem repeti-las, retornando a cidade de origem (GOLDBARG e LUNA, 2005).

Uma alternativa para a solução desse problema consiste em testar as permutações possíveis, de modo a verificar por busca exaustiva o caminho mais curto. Dado que a quantidade de permutações é o número de cidades menos um fatorial, tal solução torna-se impraticável, não sendo utilizada para tal solução.

Baseado no algoritmo Ant-Q, nós desenvolvemos um framework o qual resolve o problema do PCV. O algoritmo Ant-Q é executado da seguinte maneira:

Algoritmo Ant-Q para o PCV

```
01  Início
      Distribuir os nós na tabela;
      Calcular e distribuir o AQ0 conforme equação (1);
02    Para (cada iteração) Repita
        Definir a posição inicial dos Agentes;
03      Enquanto (existirem nós a serem visitados) Faça
04        Para (cada Agente) repita
05          Se (q0 <= rand(0..1)) Então
              Escolher a ação conforme a equação (2);
06          Senão
              Escolher a ação conforme a equação (3);
            Fimse;
            Atualizar feromônio da aresta i; (4)
          Fimpara;
        Fimenquanto;
```

```
    Calcular a melhor viagem da iteração;
    Atualização global, conforme a equação (5);
  Fimpara;
Fim.
```

Quadro 1: Pseudocódigo – Funcionamento do Algoritmo Ant-Q

O algoritmo pode ser descrito em palavras:

a) O calculo do AQ_0 (feromônio inicial) é realizado pela equação 1:

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

$$d_{ij} = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \tag{2}$$

onde *avg* apresenta a média das distâncias euclidianas dos nodos pares *ij*, calculado pela equação 2 (Teorema de Pitágoras), e *n* o número de agentes (formigas).

```
                Função calculaAQ0(numNodos, numAgentes)
01  Início
02    Para (cada par de nodos) Repita
        soma ← calcDist(x1, y1, x2, y2); // Equação 2
      Fimpara;
03    media ← soma / numNodos;
04    AQ0 ← 1 / (media * numAgentes);
05    return (AQ0);
06  Fim.
```

Quadro 2: Pseudocódigo – Calculo do AQ_0

Após calcular o valor de AQ_0, o valor é atribuído a todas as arestas que compõem o grafo.

b) A cada iteração (ciclo ou época) todos os agentes percorrem as cidades (nodos) e voltam à cidade de origem. Cada agente na iteração x realiza várias ações (ir de uma cidade para outra), guiado pelo valor do feromônio ou de uma heurística (proporcional ao inverso de sua distância). Uma importante notação do algoritmo é a forma de atualização da tabela de aprendizagem, que pode ser local, ou global. O funcionamento das atualizações é mostrado a seguir. O valor do parâmetro q0 define o tipo de exploração adotado pelo agente a cada ação. Para isso, é sorteado de maneira randômica um valor entre 0 e 1, e em seguida é comparado com o parâmetro. Caso o valor gerado seja inferior ou igual a q0, então o agente adota a ação do tipo *gulosa* (busca maior valor na ação), caso seja maior, o agente usará a estratégia exploratória.

$$s = \begin{cases} \arg\max_{u \in j_k(r)} \{[AQ(r,u)]^{\delta}.[HE(r,u)]^{\beta}\} & if\ q \le q_0 \\ S & caso\ contrário \end{cases} \quad (3)$$

Função exploitation() *// Equação 3*
01 Início
02 Para (cada aresta i, j) Repita
03 Se (j >= max) Então
max ← j;
Fimse;
Fimpara;
04 return (max);
05 Fim.

Quadro 3: Pseudocódigo – Função exploitation (busca gulosa)

$$p_k(r,s) = \begin{cases} \dfrac{[AQ(r,s)]^{\delta}.[HE(r,s)]^{\beta}}{\sum_{z \in J_k(r)} [AQ(r,z)]^{\delta}.[HE(r,z)]^{\beta}} & if\ s \in J_k(r) \\ 0 & caso\ contrário \end{cases} \quad (4)$$

Função exploration()
01 Início
02 Para (cada nodo a ser visitado) Repita
probabilidade ← calcProb(); *// Equação 4*
Fimpara;
03 Para (cada nodo a ser visitado) Repita
04 Se (probabilidade = rand(nodo)) Então
nodoEscolhido = nodo;
Fimse;
Fimpara;
05 Return (nodoEscolhido);
06 Fim.

Quadro 4: Pseudocódigo – Função exploration

Para cada ação do agente *k*, o valor na aresta é atualizado conforme a equação 5:

$$AQ(r,s) \leftarrow (1-\alpha).AQ(r,s) + \alpha.\left(\Delta AQ(r,s) + \gamma.\max_{z \in J(s)} AQ(s,z)\right) \quad (l5)$$

na qual o valor para *max* é obtido através de uma busca pelo maior valor entre todas as arestas adjacentes a *s* (nó destino), no intuito de descobrir se usará tal aresta para a boa solução.

```
Função atualizacaoLocal(i, j)
01  Início
02    max ← calcMax();
03    AQ(i, j) = calcFeromon(i, j, max); // Equação 5
04  Fim.
```

Quadro 5: Pseudocódigo – Atualização local

A atualização global pode ocorrer em duas situações: i) no final de cada iteração é selecionada a melhor rota e então realizada a atualização local com um parâmetro de reforço para cada aresta da rota; ii) sempre que uma melhor rota é encontrada, as arestas conectadas aos nodos receberão um reforço. Ambas as atualizações ocorrem da mesma forma, detalhadas a seguir:

$$\Delta AQ(r,s) = \begin{cases} \dfrac{W}{L_{Best}} \end{cases} \qquad (6)$$

no qual W representa o valor do feromônio (peso) a ser distribuído entre as arestas da melhor rota, e L_{Best} é o comprimento total da rota.

```
Função atualizacaoGlobal(melhorRota)
01  Início
02    reforço ← calcReforço(); // Equação 6
03    Para (cada aresta pertencente à melhorRota) Repita
        atualizacaoLocal(reforço); // Quadro 5
      Fimpara;
04  Fim.
```

Quadro 6: Pseudocódigo – Atualização global

Nós observamos nos experimentos iniciais que para encontrar uma política de ação sub-ótima é necessário poucas iterações. Isso ocorre devido à heurística empregada no algoritmo Ant-Q. A imagem 2 ilustra a rota de uma política de ação.

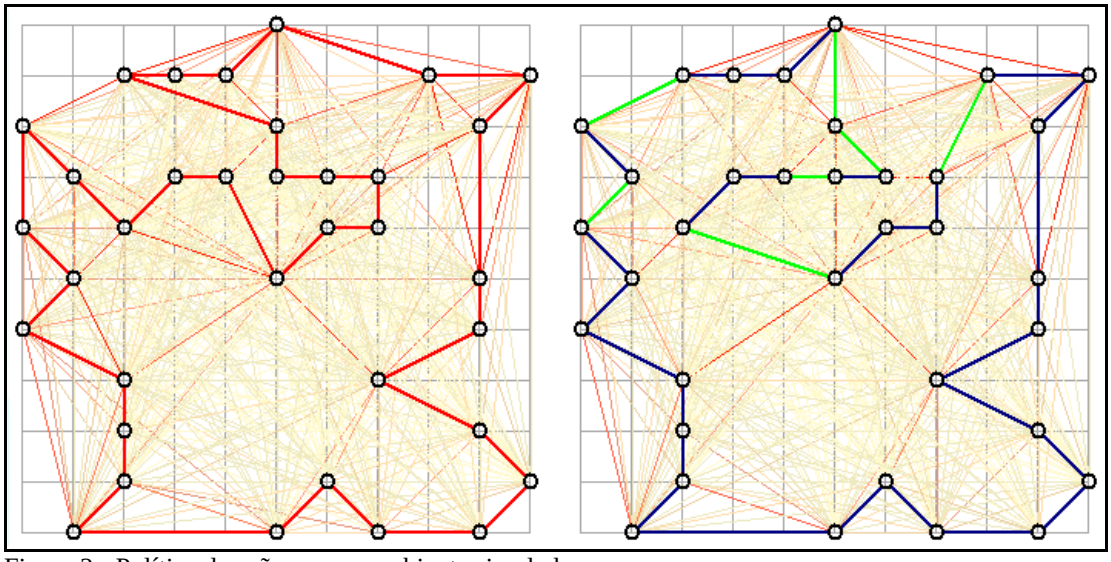

Figura 2 - Política de ação em um ambiente simulado

Podemos observar na figura 2 a direita que o ambiente é alterado após um número de iterações. Isso mostra a adaptação dos agentes em problemas de otimização.

RESULTADOS OBTIDOS

A partir da metodologia proposta, realizamos alguns experimentos preliminares que têm como objetivo verificar a coerência dos valores encontrados. Os experimentos realizados com o algoritmo Ant-Q avaliaram sua eficiência considerando fatores como: variações na taxa de aprendizagem; fator de desconto; taxa de exploração e regras de transição. Os parágrafos seguintes apresentam os valores obtidos.

Para obter os resultados da eficiência variando os parâmetros do algoritmo Ant-Q, foi utilizado três ambientes com 35, 45 e 55 estados (cidades) (Figura 3). O aprendizado em cada amostra foi rodado 5 vezes pelo algoritmo, pois se observa que fazendo experimentos em um mesmo ambiente, com fatores iguais, podem ocorrer variações na eficiência gerada pelo algoritmo. Isto ocorre porque as ações dos agentes são autônomas e os valores gerados durante sua aprendizagem são estocásticos. Portanto, as políticas de ações dos agentes podem variar de um experimento para outro. Com isso, a eficiência apresentada ao longo deste trabalho representam a média de todos os experimentos gerados com 5 repetições em cada ambiente. Esse número de repetições foi suficiente para avaliar com precisão a eficiência média do algoritmo, pois observamos que a partir deste número os resultados dos experimentos começavam a se repetir.

Observamos que alguns parâmetros como a quantidade de agentes devem ser iguais ao número de estados, conforme simulado por Dorigo e Gambardella 1996. Os seguintes parâmetros foram adotados com os seguintes valores: $\delta = 1;\ \beta = 2;\ \gamma = 0{,}3;\ \alpha = 0{,}1;\ q_0 = 0{,}9;\ W = 10$.

Foi estabelecido como critério de parada para o algoritmo o número de passos igual a 300. Dependendo do tamanho e da complexidade do ambiente, esse número não é suficiente para encontrar a melhor rota, porém o objetivo dos experimentos é testar os valores dos parâmetros utilizados e não a qualidade da política de ação.

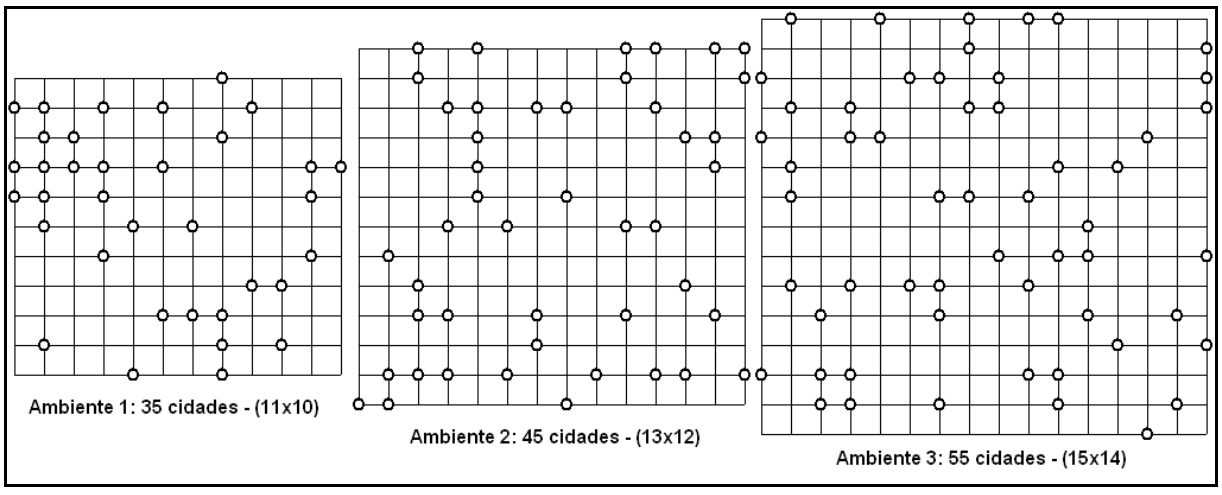

Figura 3: Ambientes utilizados nos experimentos

Os melhores valores utilizados como taxa de aprendizagem para o Algoritmo Ant-Q estão entre 0,2 e 0,3. Taxas inferiores e superiores as melhores encontradas fizeram com que o agente, ao estabelecer uma melhor ação em um determinado estado do ambiente, não efetuasse outras ações na busca por caminhos melhores. O valor 0,3 para taxa de aprendizagem foi o que apresentou melhores resultados, sendo este utilizado para os demais experimentos. Observou-se também que quanto menor a taxa de aprendizagem, menores são as variações nas amostras. Para verificar as melhores taxas foram realizados experimentos com valores entre 0 e 1 nos três ambientes apresentados anteriormente. A figura 4 apresenta a eficiência das taxas de aprendizagem.

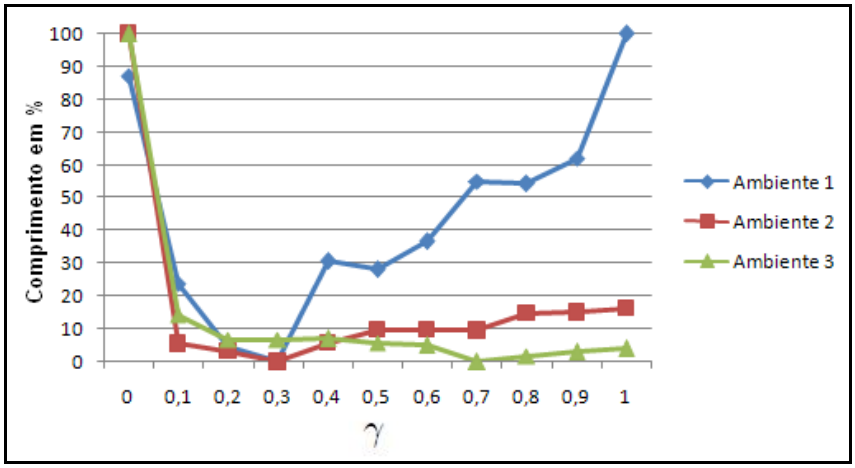

Figura 4: Resultados com Taxa de Aprendizagem (γ)

Os melhores valores para o fator de desconto estão entre 0,1 e 0,2 conforme apresentado na figura 5. Valores menores que 0,1 e maiores que 0,2 mostraram-se ineficientes para o algoritmo, tendo baixa relevância no aprendizado dos agentes, e assim estes ao invés de escolherem ações que os levem a uma política ótima de ações, escolhem ações que apenas os levam a ótimos locais.

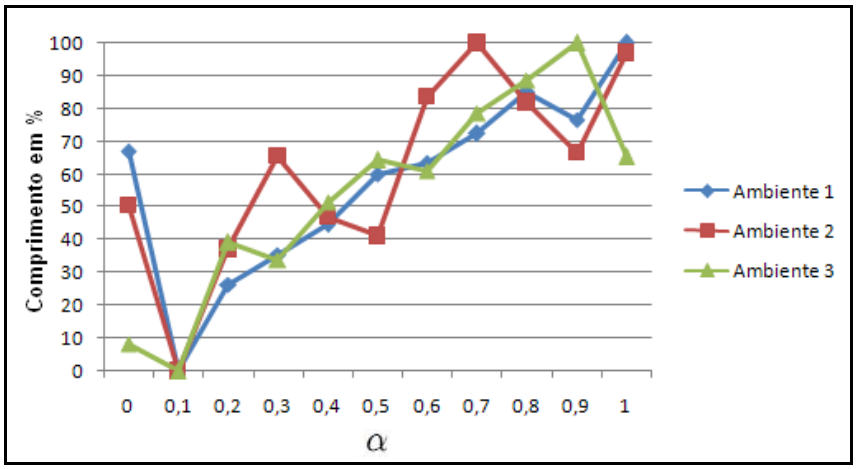

Figura 5: Resultados com Fator de Desconto (α)

Os melhores valores para Q_0 estão entre 0,8 e 1. A medida com que o valor vai se aproximando de 0, as ações dos agentes vão se tornando cada vez mais aleatórias, consequentemente as soluções começam a não serem mais interessantes.

O melhor valor para a taxa de exploração é 0,9. Com esse valor, agentes escolhem os caminhos mais curtos e que apresentam maiores concentrações de feromônio (buscas gulosas). Isso significa que 10% é a taxa de exploração para encontrar eventuais caminhos com melhores soluções.

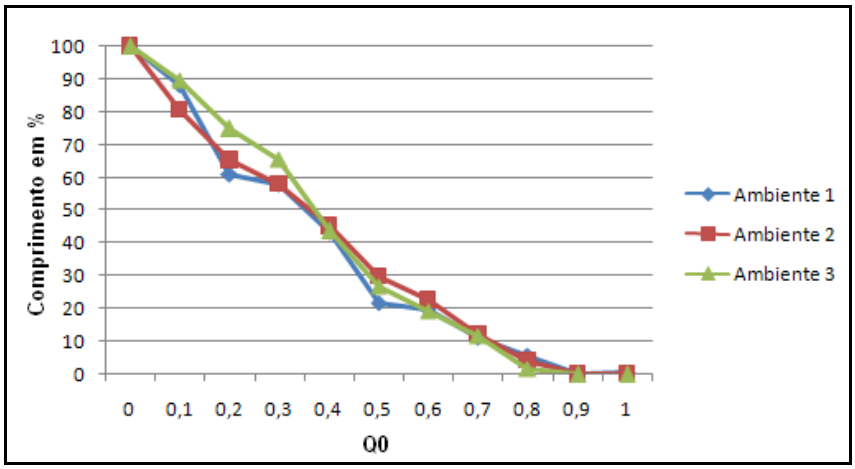

Figura 6: Resultados com Taxa de Exploração (Q0)

Os experimentos alterando os fatores δ e β foram realizados em ambientes com estados e tamanhos diferentes. Como mostra a figura 7, o algoritmo é dependente da heurística, reforçada pelo parâmetro β. Para obter bons resultados, o valor de β deve corresponder a pelo menos 65% do valor de δ.

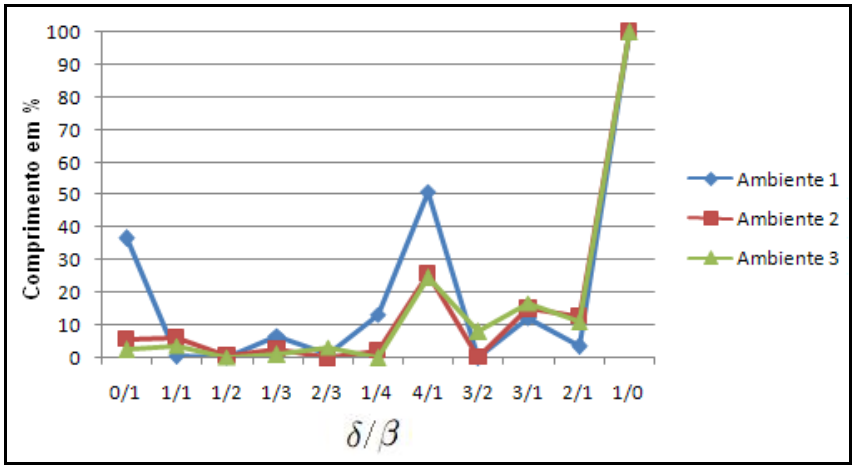

Figura 7: Resultados com Regra de Transição (δ e β)

CONSIDERAÇÕES FINAIS

A coordenação é algo implícito nas atividades humanas. Quando estas atividades são simuladas computacionalmente ou quando os sistemas devem fornecer resultados para ser aplicado em situações reais, o aspecto de coordenação influencia o processo dos agentes na busca por objetivos.

Quando se pensa na aplicação da coordenação em sistema de agentes, muitas dificuldades ainda aparecem, no sentido de especificar o que é necessário para a convergência do sistema. Isso envolve a decisão de que método de coordenação utilizar, em função dos agentes, do domínio e dos objetivos propostos; que agentes devem coordenar-se, considerando suas atividades; e a resolução de outras questões envolvidas como custo de troca de informações entre agentes e conflitos que podem surgir.

Portanto, dispor de um conjunto de critérios para analisar métodos de coordenação de agentes no projeto de um sistema beneficia o comportamento do mesmo. Também possibilita analisar a coordenação em diferentes SMA.

A aplicação de aprendizado no método de coordenação de agentes é uma nova tendência. Técnicas para aprendizagem podem ser usadas para desenvolver habilidades na solução de problemas pelos agentes. Assim, agentes podem aprender a coordenar ações, mesmo sem terem conhecimento dos outros agentes.

REFERÊNCIAS

BARAY, C. **Evolution of Coordination in Reactive Multi-Agent Systems**. PhD thesis, Computer Science Department, Indiana University, Bloomington, Indiana, 1999.

BONABEAU, E.; DORIGO, M.; THERAULAZ, G. **Swarm Intelligence: From Natural to Artificial Systems**. Oxford University Press, 1999.

BOURON, T. **Structures de communication et d'organisations pour la cooperation dans uns univers multi-agents**. Thèse de l'université Paris 6, 1992.

BULLNHEIMER, B.; HARTL, R. F.; STRAUSS, C. **An improved ant system algorithm for the vehicle routing problem**. Annals of Operations Research, 89, 1999.

CORKILL, D. D. **Hierarchical planning in a distributed environment**. In Proceeding of the Sixth International Joint Conference on Artificial Intelligence, pg. 168-175, Cambridge, Massachusetts, 1979.

COSTA, D.; HERTZ, A. **Ants can colour graphs**. Journal of the Operational Research Society 48, 295-305, 1997.

DORIGO, M. **Optimization, Learning and Natural Algorithms**. PhD thesis, Politecnico di Milano, 1992.

DORIGO, M.; GAMBARDELLA, L. M. **A Study of Some Properties of Ant-Q**. In Proceedings of PPSN Fourth International Conference on Parallel Problem Solving From Nature, pages 656-665, 1996.

DORIGO, M.; MANIEZZO, V.; COLORNI, A. **Ant System: Optimization by a Colony of Cooperating Agents**. IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1):29-41, 1996.

DURFEE, E. H.; LESSER, V. R. **Partial Global Planning: A coordination framework for distributed hypothesis formation**. IEEE Transactions on Systems, Man, and Cybernetics, 21(5):1167-1183, 1991.

DURFEE, E. H. **Distributed Problem Solving and Planning**. Chapter 3 in Gerhard Weiss, editor. Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, MIT Press, Cambridge MA, 1999.

FERBER, J. **Multi-Agent System: An Introduction to Distributed Artificial Intelligence**. Addison-Wesley, Longman Ink., New York, 1999.

GAMBARDELLA, L. M.; DORIGO, M. **Ant-Q: A Reinforcement Learning Approach to the TSP**. In proceedings of ML-95, Twelfth International Conference on Machine Learning, pp. 252-260, 1995.

GAMBARDELLA, L. M.; TAILLARD, E. D.; DORIGO, M. **Ant Colonies for the QAP**. Technical report, IDSIA, Lugano, Switzerland, 1997.

GAMBARDELLA, L. M.; MONTEMANNI, R.; RIZZOLI, A.; DONATI, A. **Ant Colony System for a Dynamic Vehicle Routing Problem**. Journal of Combinatorial Optimization, v. 10(4), pg. 327-343(17), 2005.

GOLDBARG, M.; LUNA, H. P. L. **Otimização combinatorial e programação linear: modelos e algoritmos**. Rio de Janeiro: Campus, 2005.

GOSS, S.; ARON, S.; DENEUBOURG, J. L.; PASTEELS, J. M. **Self-organized shortcuts in the Argentine ant**. Naturwissenschaften, v. 76, p. 579-581, 1989.

KETCHPEL, S. **Coalition Formation among Autonomous Agents**. Proc. European Workshop Modeling Autonomous Agents in a Multiagent World(MAAMAW 93), Springer-Verlag, Heidelberg, Germany, pg. 73-88, 1993.

LESSER, V. R.; ORTIZ, C. L.; TAMBE, M. **Distributed Sensor Networks: a Multiagent Perspective**. Massachusetts, New York: Kluwer Academic Publishers, v. 9, 386 p., 2003.

OSSOWSKI, S. **Coordination in Artificial Agent Societies (Social Structure and Its Implications for Autonomous Problem-Solving Agents)**. Berlin: Springer (Lecture Notes in Artificial Intelligence 1535), 1999.

ROUX, O.; FONLUPT, C.; TALBI, E-G.; ROBILLIARD, D. **ANTabu – Enhanced Version**. Technical Report LIL-99-01, Laboratoire d'Informatique du Littoral Université du Littoral, Calais, France, 1999.

SMITH, R. G.; DAVIS, R. **Negotiation as a metaphor for distributed problem solving**. Artificial Intelligence, 20:63-109, 1983.

STÜTZLE, T.; HOOS, H. **MAX-MIN Ant System and Local Search for The Traveling Salesman Problem**. In Proceedings of the IEEE International Conference on Evolutionary Computation, pp. 309-314, 1997.

WOOLDRIDGE, M. J. **An Introduction to MultiAgent Systems**. John Wiley and Sons, 2002.

[1] Projeto de Pesquisa – FAP (SEDEP) – Universidade do Contestado, Mafra, 2009

[2] Bacharelando em Sistemas de Informação. UnC Mafra - SC, ronszcka@gmail.com;
[3] Bacharelando em Sistemas de Informação. UnC Mafra - SC; mariogjro@gmail.com;
[4]Mestre em Informática Aplicada, docente do curso de Sistemas de Informação, UnC – Mafra. prof.richard@gmail.com.