Problema Ant-based Clustering: Algoritmo de solução

André Nunes[1], **Rafael Stubs Parpinelli**[2]

[1]Estudante de Ciência da Computação – Universidade Do Estado de Santa Catarina (UDESC)

[2]Professor de Ciência da Computação – Universidade Do Estado de Santa Catarina (UDESC)

dekonunesss@gmail.com, rafael.parpinelli@udesc.br

Resumo. *Com o crescimento da computação os algoritmos utilizados no passado começaram a ficar defasados e, por consequência, não serem mais viáveis para o uso no dias de hoje. Atualmente no campo da Inteligência Artificial buscam-se soluções baseadas na natureza. A simulação apresentada é feita em um algoritmo de colonização de formigas, onde com movimentos aleatórios e tomadas de decisões baseadas em estatísticas os itens espalhados em uma matriz começam a criar aglomerados.*

1. Introdução

Com os avanços da computação a quantidade de dados aumentou significativamente, em decorrência disso houve o surgimento de novas pesquisas, uma delas é a de Data Mining, que consiste em retirar conhecimento de uma grande quantidade de dados [Jafar and Sivakumar 2010], para tal fim existem inúmero tipos de algoritmos. A pesquisa de tipo Data Mining não é usada apenas na área da computação, como também na área da sociologia, psicologia, comércio, biologia, entre outros [Handl et al. 2003]. Será apresentado no presente artigo um modelo de Data Mining baseado na maneira que as formigas organizam seus formigueiros. O artigo visa a elaboração de um algoritmo de agrupamento baseado em ant-based clustering com os testes sendo comparados com a mudança no raio de visão dos agentes.

O artigo é composto da problemática de ant-based clustering na seção 2, o algoritmo criado e o detalhamento do mesmo na seção 3, os testes e análises sendo discutidos na seção 4 e a conclusão do artigo na última seção.

2. Problema Ant-based Clustering

As formigas da classe operária fazem a limpeza das formigas mortas na colônia agrupando-as e criando cemitérios. O movimento das formigas (agentes) é aleatório e sua decisão de pegar ou largar um item também é realizada de forma aleatória, porém com a influência de seu raio de visão e dos itens ao seu redor. A probabilidade do agente deixar o objeto (dados) é acrescida a cada item que há ao seu redor, assim como a probabilidade de pegar um item é acrescida quando há poucos ou ainda nenhum item em seu campo de visão [Jafar and Sivakumar 2010].

O principal objetivo do algoritmo é agrupar dados de maneira parecida com a que as formigas fazem para agrupar as formigas mortas nas colônias.

2.1. PEAS

PEAS (Performance, Environment, Actuators, Sensors) é uma ótima estratégia para facilitar o entendimento do problema e detalhar melhor as partes para que se consiga criar um algoritmo menos propenso a falhas.

2.1.1. Características do ambiente

- Agentes: Formigas (agentes que podem se mover);
- Medida de desempenho: velocidade das formigas, quantidade de objetos carregados e máximo de itens agrupados;
- Ambiente: Recipiente virtual para o deslocamento dos agentes;
- Sensores: Antenas, patas, vizinhança;
- Atuadores: Braço das formigas, movimento, tomada de decisão.

2.1.2. Propriedades do ambiente

- Ambiente de tarefa: Matriz;
- Observável parcialmente: A formiga (agente) tem apenas a visão do seu raio de visão e não da matriz toda;
- Estocástico: O movimento das formigas (agentes) é totalmente aleatório;
- Sequencial: O item colocado em um determinado local influencia no resultado de outra formiga;
- Dinâmico: As formigas mudam o ambiente pegando ou deixando um item;
- Discreto: A quantidade dos dados, formigas, itens e matriz não se altera;
- Multiagente: Existe mais de um agente no ambiente.

3. Algoritmo Implementado

O algoritmo para a resolução do problema Ant-based Clustering pode ser implementado de diferentes maneiras, podendo ser simples ou com várias inteligencias sobre o algoritmo. O algoritmo criado no presente trabalho contém algumas inteligências, como a de que cada formiga pode ter seu próprio raio de visão, bem como no final das interações todas as formigas continuam até localizar um local para deixar os itens, por exemplo.

Toda a lógica se baseia em uma matriz, nela ficam representados os itens e as formigas que são modificados durante o processo e usados para os algoritmos de decisão.

A tecnologia usada foi a lingaguem C++ e a biblioteca Simple and Fast Multimedia Library (SFML-2.3.1) para a interface gráfica.

3.1. Implementação

Como descrito por [Handl et al. 2003] a matriz é povoada randomicamente, tanto as formigas quanto os itens nela espalhados, com o objetivo de facilitar a visualização, as células são diferenciadas por cores distintas, de modo que toda célula de cor branca significa que não há nada nela, as células de cor **azul** representam um item, as células de cor **vermelha** representam que há uma formiga sem item; as células de cor **preto** representa uma formiga com item; as células de cor **verde** representa que há uma formiga sem

possuir item e que não há nenhum item abaixo dela; quando as células forem **amarelas** representa que há uma formiga carregando um item com outro item abaixo dela. O movimento das formigas pode ser para qualquer lado, sendo possível para cima ou para baixo, para a direita ou para a esquerda, bem como para suas diagonais. O movimento é aleatório, tendo como única restrição de movimento o caso onde o local escolhido já tenha outra formiga.

A probabilidade de uma formiga pegar um item é a mesma que uma formiga tem de deixar um item, porém de maneiras opostas. Essa decisão é tomada dentro de uma análise estatística onde a formiga analisa dentro do seu raio de visão todas as células e guarda em sua variável o valor de células que foi possível analisar e a quantidade de itens que estão nessas células analisadas. Após, é aplicada a fórmula $P_p = (k/k + f)^2$ onde o P representa a probabilidade, k a quantidade de itens no raio de visão e f a quantidade de células que a formiga consegue observar [Jafar and Sivakumar 2010]. Após a aplicação da fórmula é gerado um número aleatório de 1 a 100, caso a formiga esteja carregando um item e esteja sobre uma célula em branco ela compara se o número gerado aleatoriamente é menor que o valor de probabilidade calculado na fórmula, caso verdadeiro, o item é colocado na célula e a formiga muda seu estado de carregando um item para vazia, o oposto ocorre quando a formiga esta sem um item e esta sobre uma célula com item. Todos os cálculos são realizados novamente para saber se o local que o item se encontra é um local ideal (próximo de outros itens), caso não seja um local propício ela pega o item e continua sua busca para localizar um local melhor.

Para se chegar ao resultado final todas as formigas que não estejam carregando um item são retiradas para que não atrapalhem as que ainda não colocaram seus itens. O sistema só finaliza após todas as formigas terem colocado os itens nos locais encontrados por elas.

3.2. Software

Os parâmetros para o sistema são:

- Quantidade de linhas;
- Quantidade de colunas;
- Quantidade de formigas;
- Quantidade de itens;
- Quantidade de interações;
- Quantidade máxima do raio.

O raio das formigas é heterogêneo, variando de um até o número escolhido no parâmetro, porém para os testes do algoritmo do presente artigo houve uma adaptação para que os agentes fossem homogêneos e com isso fosse possível uma melhor comparação nos resultados.

4. Simulações e Resultados

Todas as simulações tiveram os mesmos parâmetros, mudando apenas o valor do raio. Os testes foram realizados usando uma matriz 40x40 com 100 formigas, 250 itens e 5000000 interações. A comparação entre os testes foi feita de maneira visual entre as imagens inicias e finais dos testes.

4.1. Raio 1

O experimento de Raio 1 obteve um agrupamento melhor que os demais, criando pequenos aglomerados. Ao final não teve grandes lacunas, como houve no experimento com raio 5. Seu tempo de processamento foi rápido em comparação com os outros testes.

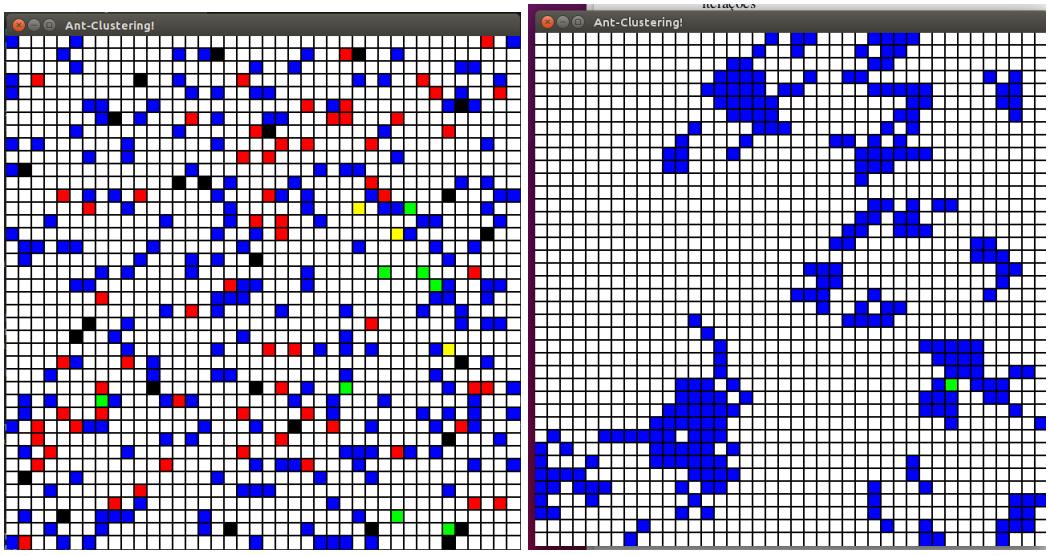

Figura 1. Raio 1 - inicial e final

4.2. Raio 2

O experimento de Raio 2 obteve um agrupamento mediano, mas ainda assim com um bom resultado satisfatório. Originou aglomerados com as bordas mais espaçadas e dispersos que o experimento com Raio 1.

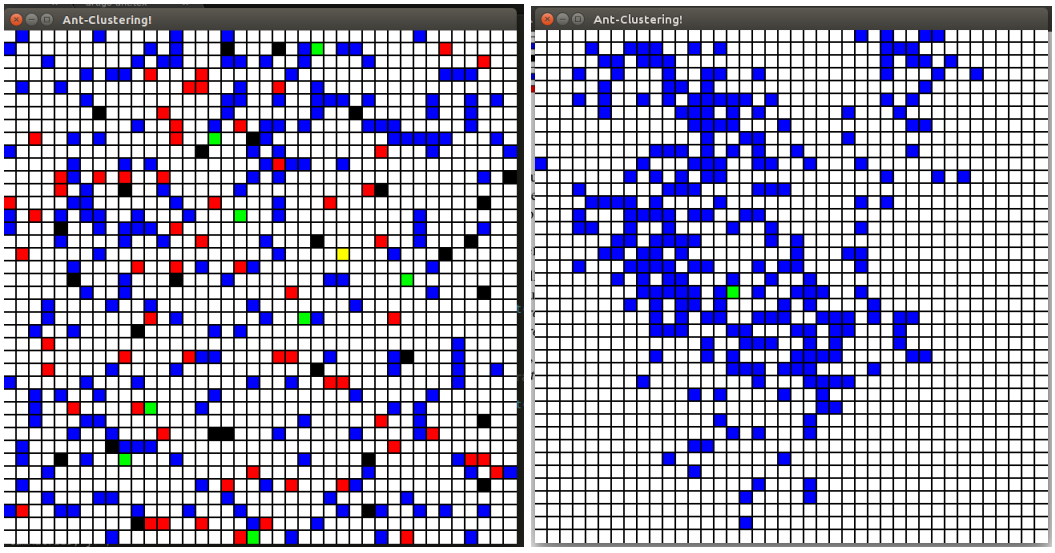

Figura 2. Raio 2 - inicial e final

4.3. Raio 5

A simulação com o resultado menos satisfatório em comparação aos outros experimentos, apesar de ter um bom agrupamento alguns itens ficaram distantes dos aglomerados devido ao valor do raio, a distancia grande de detecção de um item possibilita um item ficar bem fora do aglomerado.

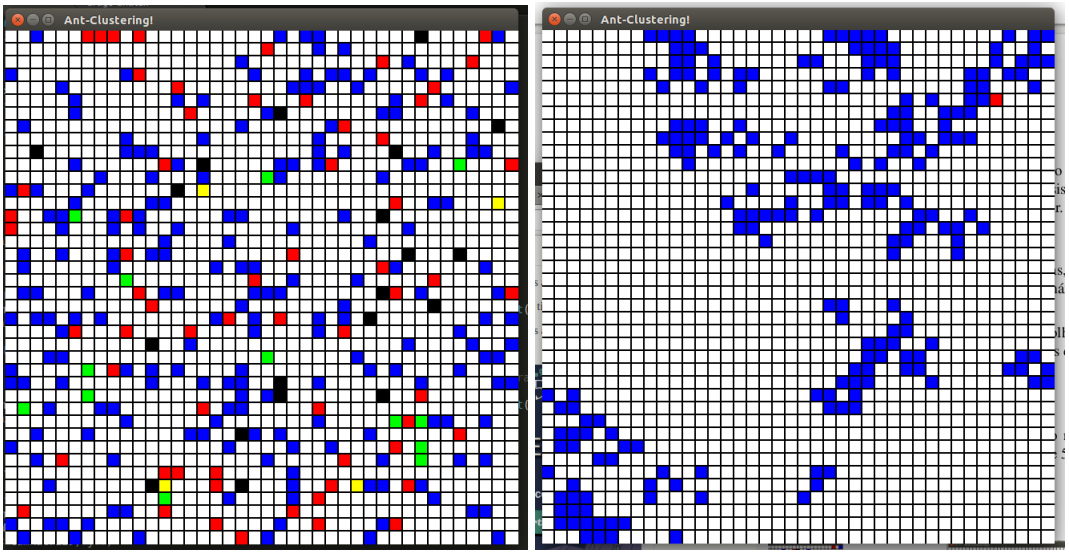

Figura 3. Raio 5 - inicial e final

5. Conclusão

O algoritmo demonstra que essa técnica pode ser aplicada para agrupamentos com um resultado satisfatório, podendo ser aplicado na área de mineração de dados para facilitar as buscas de informações.

Todas as simulações obtiveram um bom resultado, mas com um declinio a cada incremento do raio, o tempo de processamento mudava consideravelmente com o aumento do valor do raio no algoritmo aplicado.

Referências

Handl, J., Knowles, J., and Dorigo, M. (2003). Ant-based clustering: a comparative study of its relative performance with respect to k-means, average link and id-som. In *Proceedings of the Third International Conference on Hybrid Intelligent Systems, IOS Press*.

Jafar, O. M. and Sivakumar, R. (2010). Ant-based clustering algorithms: A brief survey. *International Journal of Computer Theory and Engineering*, 2(5):1793–8201.