IAR0001 - 2017/1
Relatório Trabalho 1
Ant Clustering

Alexandre Maros[1]

[1]Departamento de Ciência da Computação – Universidade do Estado de Santa Catarina
Centro de Ciências Tecnológicas – Joinville – SC – Brasil

alehstk@gmail.com

Resumo. *Ant-clustering é uma técnica de agrupamento baseado na noção de inteligência de exames. Tal conceito é amplamente utilizado para resolver problemas numéricos e combinatoriais e vem apresentando bons resultados. Este relatório pretende explorar um comportamento específico de formigas que tem como objetivo final o agrupamento de formigas mortas para liberar espaço do ambiente. Aqui será visto alguns conceitos, onde isso é aplicado, como isso é aplicado e alguns testes baseado no algorítimo aqui construído.*

1. Introdução

O agrupamento baseado em formigas (*Ant-clustering*) é uma técnica de agrupamento bioinspirada baseado na noção de inteligência de enxame [Abraham et al. 2006] que é um paradigma de inteligência distribuída para resolver problemas de otimização que surgiram com o estudo colônias, enxames ou organismos sociais. Agrupamento com algorítimos de inteligência de enxames vem sendo amplamente utilizados para resolver problemas numéricos e combinatoriais [Jafar and Sivakumar 2010] e vem crescendo bastante pois vem mostrando produzir bons resultados nessas aplicações.

Formigas pertencem a um grupo social. Sozinhas elas não conseguem sobreviver por muito tempo. Entretanto, quando convivendo em um grande grupo, algumas padrões interessantes são observados, como por exemplo quando elas estão a procura de formiga. Primeiramente, diversas formigas procuram aleatoriamente no terreno em que elas se encontram, deixando um rastro de feromônios. Essa substância é notada por outras formigas que ajudam em sua próxima decisão (devo ir para um caminho já visitado?). Com o passar do tempo, uma solução incremental é construída. A quantidade de feromônios em um determinado caminho equivale ao número de formigas que passaram por lá. As comunicações entre as formigas não são diretas, mas sim através de modificações do ambiente [Jafar and Sivakumar 2010].

O comportamento estudado nesse artigo é o de organização de itens dispostos em um espaço físico. Como observado na Figura 1, quando formigas vivas são postas em um ambiente com formigas mortas é possível notar que conforme o tempo passa as formigas vivas agrupam as formigas mortas para liberar espaço para movimento.

Utilizando esse comportamento, é possível aplicar a mesma lógica para gerar agrupamentos de determinados tipos de dados que estão dispostos em uma matriz de duas dimensões, como por exemplo mineração de dados [Abraham and Ramos 2003], recuperação de dados, organização de documentos [Ramos and Merelo 2004], análise de rede [Ekola et al. 2004], entre outros.

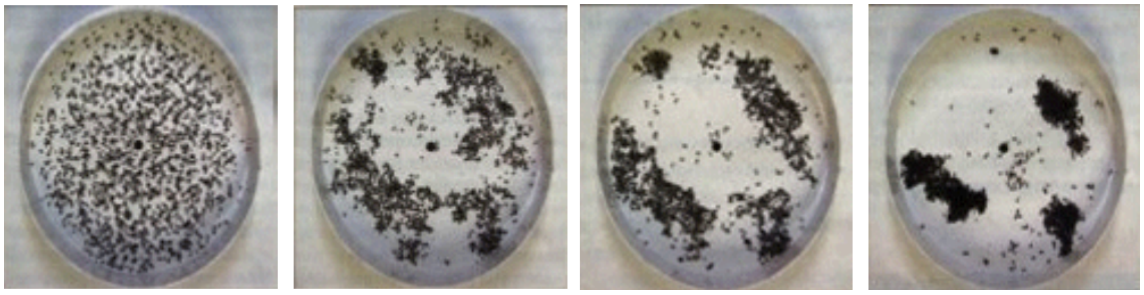

Figura 1. Agrupamento de formigas mortas
[Jafar and Sivakumar 2010]

Esse relatório estudará como esse algorítimo pode ser implementado e apresentará alguns resultados com base nos algorítimos construídos.

2. Problemática

O algorítimo segue alguns princípios básicos: as formigas são modeladas por agente simples que se movem aleatoriamente em um ambiente fechado. Formigas mortas são espalhadas nesse ambiente também de forma aleatória. Tais formigas mortas podem ser pegas e carregadas por formigas vivas dado uma função de probabilidade para cada ação (pegar e soltar). Tal função é feita se baseando na densidade de formigas mortas em volta da formiga viva. Quanto menos formigas mortas ao redor da formiga viva, mais chances aquela formiga terá de pegar uma morta e carregar para alguma direção. Quanto mais formigas mortas estiverem no raio de observação da formiga viva, maior a chance dela deixar a formiga morta naquela posição.

2.1. PEAS

- **Medida de desempenho**
  - Maximizar densidade de formigas em um local;
  - Maximização do agrupamento (Poucos grupos com muitas formigas).
- **Ambiente**
  - Formigas vivas e mortas;
  - Placa de Petri (tabuleiro).
- **Sensores**
  - Reconhecer a existência de uma formiga em suas proximidades;
  - Verificar se a formiga está morta.
- **Atuadores**
  - Mover;
  - Soltar formiga;
  - Carregar formiga.

2.2. Propriedades do ambiente

- **Parcialmente observável**: cada formiga tem um campo de visão limitado;
- **Estocástico**: há um componente aleatório, fazendo com que cada execução gere um resultado diferente;
- **Sequencial**: as decisões atuais afetam as decisões futuras;

- **Estático**[1]: uma formiga não pode afetar o ambiente enquanto outra está atuando;
- **Discreto**[2]: Todos os elementos do ambiente são contáveis;
- **Multiagente**: Há diversas formigas atuando sobre o mesmo ambiente.

3. Modelo implementado

O trabalho foi implementado utilizando a linguagem C++ e a biblioteca gráfica SFML (*Simple and Fast Multimedia Library*). A implementação pode ser divida em duas grandes partes, o *loop* principal (Algorítimo 1) e a função de atualização das formigas (Algorítimo 2).

Algorítimo 1. Loop principal

```
1   Cria tabuleiro n × n
2   Popule o tabuleiro com x formigas mortas
3   Popule o tabuleiro com y formigas vivas
4   Enquanto não calculou todas as iterações:
5   |   Para cada formiga viva:
6   |   |   Atualize seus movimentos
```

Algorítimo 2. Atualização da Formiga

```
1   Caso a formiga não esteja carregando uma morta:
2   |   Se há uma formiga morta na posição atual:
3   |   |   Calcule a probabilidade de carregar com a Fórmula 1
4   |   |   Sorteie um inteiro aleatorio entre 0 e 100
5   |   |   Se o número for menor ou igual a probabilidade:
6   |   |   |   Pegue a formiga morta
7   Caso a formiga já esteja carregando uma morta:
8   |   Caso esteja em um espaço vazio:
9   |   |   Calcule a probabilidade de soltar com a Fórmula 2
10  |   |   Sorteie um inteiro aleatorio entre 0 e 100
11  |   |   Se o número for menor ou igual a probabilidade:
12  |   |   |   Solte a formiga
13
14  Mova para uma posição aleatória
```

Após a inicialização e população do tabuleiro, será executada um número de iterações fixas, sendo que uma iteração corresponde ao movimento de todas as formigas vivas no tabuleiro. Como o desenho do tabuleiro é algo demorado, é possível configurar quantas interações são necessárias para que o tabuleiro seja redesenhado.

Da forma com que foi implementada, a ação das formigas são sequenciais, isto é, uma formiga age apenas após a outra completar seu movimento, sem que aja sobreposição de ações. Isso ocorre pois não há a utilização de processamento paralelo (*threads*). As

[1] O trabalho não foi implementado com *Threads*, caso contrário seria dinâmico

[2] Na vida real seria contínuo já que a movimentação é contínua e não discreta, entretanto, aqui estamos lidando com uma grid de posições finitas

formigas podem se mover na diagonal, dessa forma existem 8 possíveis direções de movimento. Não há nenhuma função especial de movimento, sendo que apenas um gerador aleatório decide a próxima direção, sem levar em consideração outros fatores.

Outro ponto importante para notar é que não há bordas no tabuleiro, ou seja, uma formiga que está no extremo direito do tabuleiro, ao andar uma casa para a direita, chegará na posição extrema esquerda, como se o tabuleiro fosse um globo.

A fórmula para decidir se a formiga deve começar a carregar uma formiga morta é definida da seguinte forma:

$$P_c = \begin{cases} \left(\frac{n_{vazios}}{n_{total}}\right)^2 \times 100, & \text{se } n_{vazios} \neq n_{total} \\ 99, & \text{se } n_{vazios} = n_{total} \\ 1, & \text{se } n_{vazios} = 0 \end{cases} \tag{1}$$

onde:

- P_c é a probabilidade da formiga carregar a formiga morta encontrada;
- n_{vazios} é o número de espaços observados sem formigas mortas;
- n_{total} é o número total de espaços observados pela formiga

Já a fórmula para decidir se a formiga deve soltar a formiga morta que esta carregando é dada por:

$$P_s = \begin{cases} \left(\frac{n_{mortas}}{n_{total}}\right)^2 \times 100, & \text{se } n_{mortas} \neq n_{total} \\ 99, & \text{se } n_{mortas} = n_{total} \\ 1, & \text{se } n_{mortas} = 0 \end{cases} \tag{2}$$

onde:

- P_s é a probabilidade da formiga soltar a formiga morta sendo carregada;
- n_{mortas} é o número de espaços observados com formigas mortas;
- n_{total} é o número total de espaços observados pela formiga

4. Experimentos, resultados e análises

Os experimentos foram realizados com os seguintes parâmetros:

- Tabuleiro com tamanho 160×160;
- 6000 formigas mortas;
- 100 formigas vivas;
- 4 milhões de iterações;
- Raio de visão variando entre 1, 5 e 10.

O primeiro experimento realizado identificado pela Figura 2 mostra o agrupamento das formigas com sementes para o algorítimo de geração de números aleatórios baseados no tempo em que está sendo executado o experimento. A mesma semente foi utilizada para gerar o tabuleiro inicial, logo a disposição inicial das formigas são iguais para todas as variações de raio, porém os movimentos são diferentes de uma execução para outra. Com raio 1 foram realizados 11 agrupamentos, com raio 5 foi observado 4 agrupamentos e com raio 10 surgiram 3 agrupamentos.

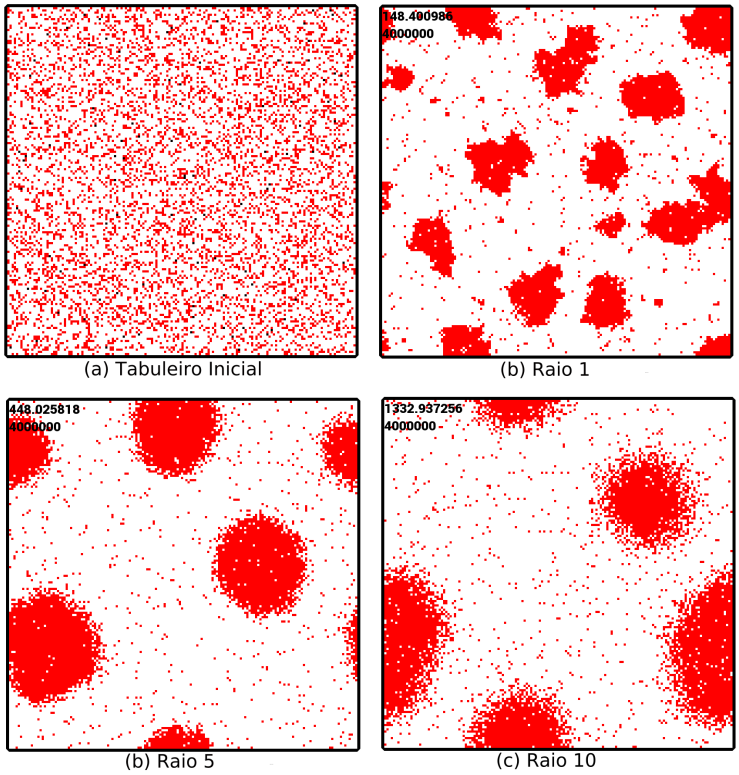

Figura 2. Experimento 1 – Agrupamento de formigas com sementes diferentes.

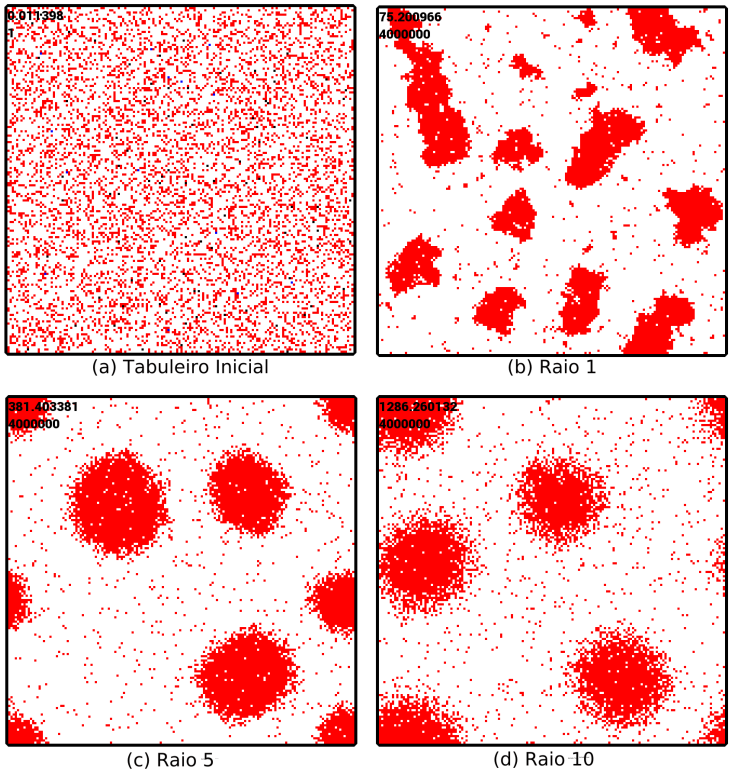

Figura 3. Experimento 2 – Agrupamento de formigas com a mesma semente.

O segundo experimento identificado pela Figura 3 mostra o agrupamento realizado pelas formigas com a mesma semente. Esse experimento foi realizado pois mostra como o raio afeta o agrupamento dado uma mesma configuração inicial e os mesmos movimentos. As formigas com raio 1 realizaram 9 agrupamentos, com raio 5 houve 5 agrupamentos e com raio 10 houve 4 agrupamentos.

É possível observar que quanto maior o raio menos grupos com mais formigas foram formadas, entretanto, mais formigas "soltas", isto é, fora de um grupo são observadas. Isso é explicado pois quanto maior o raio, maiores são as chances de uma formiga viva pegar uma morta que está na borda do agrupamento. Outro fator que ocorre ao aumentar o raio é o aumento de buracos no meio dos agrupamentos. Com um raio de 10, a formiga viva que estará no meio do agrupamento ainda conseguirá observar fora do agrupamento e identificará que há espaços vazios em volta, aumentando a chance da formiga pegar a morta.

Um ponto interessante é a velocidade com que os grupos são formados. A Figura 4 mostra o estado do sistema na iteração 100000 (cem mil). Quanto menor o raio, mais rápido fica evidente a formação dos grupos. Nessa iteração, quando as formigas estão utilizando Raio 1 já fica claro o agrupamento. Em contrapartida, quando utilizando Raio 10, os agrupamentos são tão esparsos que parece estar em um ambiente completamente aleatório.

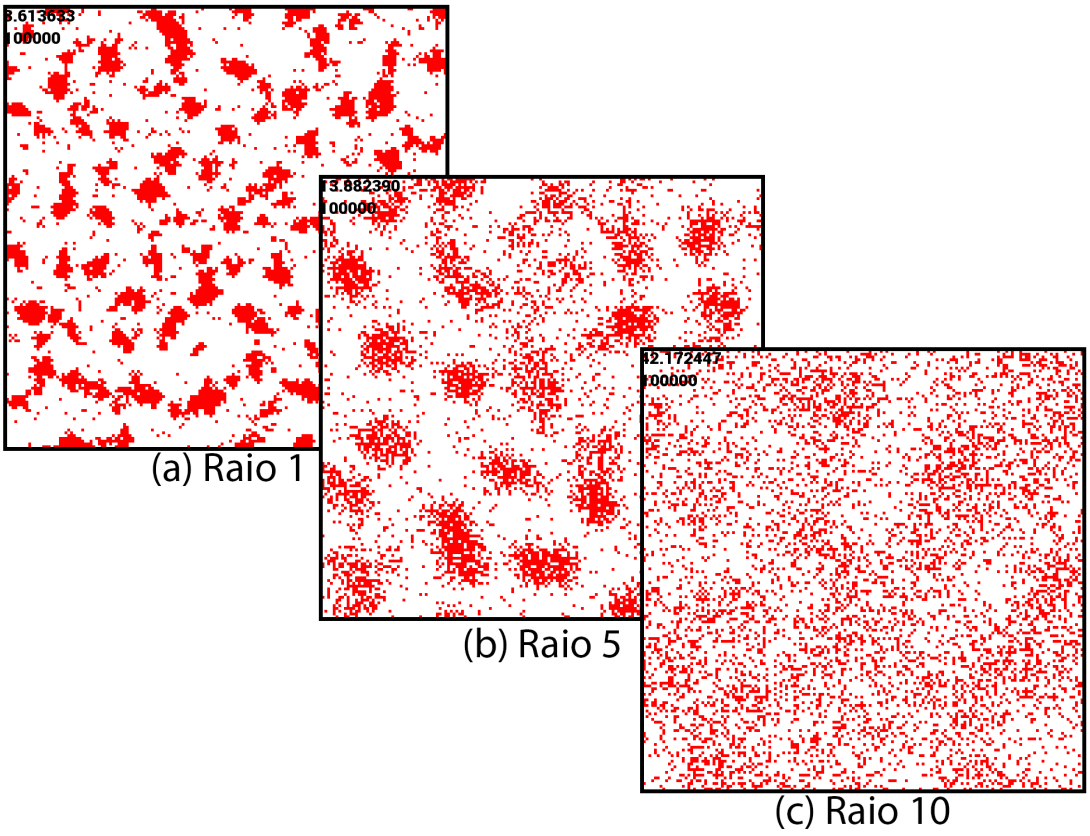

Figura 4. O estado do sistema na iteração 100000 (cem mil) para os raios 1, 5 e 10. Quanto menor o raio, mais rápido grupos são formados.

Quanto menor o raio, mais grupos são formados, porém eles são mais concisos, densos. Quanto maior o raio, menos grupos com mais formigas são formados, porém eles acabam ficando com diversos buracos no meio e com mais formigas espalhadas pelo tabuleiro.

Outro fator de desempenho observado é o tempo. O primeiro valor de cada quadrante das Figuras 2 e 3 é o tempo necessário para todas as iterações serem concluídas.

Tempo (segundos) para calcular 4 milhões de iterações		
	Experimento 1	Experimento 2
Raio 1	148.400s	75.200s
Raio 5	448.025s	381.403s
Raio 10	1332.937s	1286.260s

5. Conclusão

Os resultados demonstraram que o algorítimo estudado é realmente interessante para realizar problemas de agrupamento. Mesmo em um ambiente simples, com fórmulas probabilísticas descomplicadas e movimentos completamente aleatórios o resultado produzido se mostrou bastante eficaz.

Há três grandes tópicos a serem explorados em trabalhos futuros:

1. Paralelizar a ação das formigas;
2. Estabelecer uma melhor função de movimento;
3. Melhorar as fórmulas de probabilidade estabelecidas.

Abordando esses 3 tópicos, é esperado que a eficácia do sistema seja ainda maior.

Referências

Abraham, A., Guo, H., and Liu, H. (2006). Swarm intelligence: foundations, perspectives and applications. In *Swarm Intelligent Systems*, pages 3–25. Springer.

Abraham, A. and Ramos, V. (2003). Web usage mining using artificial ant colony clustering and linear genetic programming. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 2, pages 1384–1391. IEEE.

Ekola, T., Laurikkala, M., Lehto, T., and Koivisto, H. (2004). Network traffic analysis using clustering ants. In *Proceedings World Automation Congress, 2004.*, volume 17, pages 275–280.

Jafar, O. M. and Sivakumar, R. (2010). Ant-based clustering algorithms: A brief survey. *International journal of computer theory and engineering*, 2(5):787.

Ramos, V. and Merelo, J. J. (2004). Self-organized stigmergic document maps: Environment as a mechanism for context learning. *arXiv preprint cs/0412075*.