Trabalho Prático II - ACO

Daniel Carneiro
Universidade Federal de Minas Gerais
Belo Horizonte, MG, 31.270-901
dennys@ufmg.br

Abstract

*O objetivo desse trabalho foi a implementação de um algoritmo de **Ant Colony Optimization** (ACO) para o **longest path problem**. O algoritmo foi implementado em python* 3.10 *e utiliza a biblioteca externa numpy.*

1. Introduction

Ant Colony Optimization (ACO) é uma metaheurística utilizada para resolver problemas que podem ser reduzidos a encontrar bons caminhos em grafos. Nessa metaheurística utilizamos formigas artificiais para caminhar no gráfico e, imitando o comportamento de formigas reais, deixar feromônios no melhor caminho para guiar (probabilisticamente) as outras formigas para caminhos bons.

Nesse trabalho foi estudado e implementado um algoritmo de ACO com o objetivo de aproximar uma solução para o **longest path problem**, no qual o objetivo é encontrar o maior caminho em um grafo entre quaisquer dois vértices. Nas próximas sessões iremos descrever as decisões de implementação, otimização de parâmetros, experimentos e resultados.

2. Decisões de Implementação

A versão de ACO implementada foi a $\mathcal{MMAS}$ (Max Min Ant System) 1, utilizando o descrito no livro *Ant colony optimization* (Dorigo and Stutzle)[1]. As decisões tomadas foram descritas a seguir.

2.1. Paralelismo

O algoritmo 1 mostra como que o paralelismo foi implementado, nós construímos os caminhos em paralelo, e atualizamos os feromônios de maneira sequencial.

2.2. Representação da Solução

A solução é representada pela sequência de vértices que visitamos (e.g. $v_2v_1v_3$).

Algorithm 1 ACO

$t \leftarrow 1$
best $\leftarrow$ (0, [])
Initialize graph with t_{max} pheromones on every edge
while $t <$ maxIt **do**
  Build popSize solutions in parallel $\rightarrow S$
  itbest $\leftarrow (\max S.\text{length}, \max S.\text{path})$
  if itbest[0] $>$ best[0] **then**
    best $\leftarrow$ itbest
  end if
  $r \leftarrow$ Choose from {best, itbest} with probabilities $(t/\text{maxIt}, 1 - t/\text{maxIt})$
  Update pheromones with r.path
  $t \leftarrow t + 1$
end while
return best

2.3. Probabilidade de Transição

A probabilidade de transição por uma aresta é dada pela equação

$$\frac{w^\alpha n^\beta}{\sum_{v_i \in N} w_i^\alpha n_i^\beta}$$

Onde w é o feromônio da aresta, n é o peso, N é a vizinhança, e n_i e w_i são o peso e o feromônio da aresta que conecta o vértice atual a i.

2.4. Atualização dos Feromônios

O $\mathcal{MMAS}$ têm como característica a utilização de um limite mínimo e máximo para os feromônios, nesse trabalho foi utilizado um limite variável onde t_{max} é o tamanho do caminho sendo utilizado para atualizar os feromônios dividido pela taxa de evaporação e t_{min} é $t_{max} \cdot ((1 - \sqrt[n]{0.05})/((avg - 1)\sqrt[n]{0.05}))$ onde n é o número de vértices do grafo e avg é a média de arestas que saem de um vértice.

Como explicitado no algoritmo 1, no inicio das iterações nós atualizamos o feromônio utilizando com maior probabilidade o melhor caminho encontrado naquela iteração

para promover exploração, e depois se torna mais provável que utilizemos o melhor caminho encontrado até então para promover o exploitation.

2.5. Parâmetros

Os parâmetros ajustáveis pelo usuário são o número de iterações, o número de formigas, α, β e a taxa de evaporação ρ. Esses parâmetros foram ajustados para o problema de caminho mais longo na sessão a seguir.

3. Otimização de Parâmetros

Os parâmetros iniciais utilizados foram os da tabela 1 como sugerido por Dorigo and Stutzle[1]. Todos os parâmetros foram ajustados utilizando o arquivo de testes *entrada3.txt*, por possuir o maior número de caminhos possíveis e, portanto, maior variedade de soluções.

Em todos os testes foram realizadas 30 execuções, e todos os dados representados nos gráficos são a média dos resultados obtidos nas 30 execuções.

Table 1. Parâmetros iniciais, ρ é a taxa de evaporação, C^{fn} é o custo da solução obtida pela heurística gulosa de vizinho mais distante.

Parâmetro	Valor
Feromônio Inicial	$1/\rho C^{fn}$
Número de Formigas	10
β	5
ρ	0.02
α	1
Número de iterações	400

3.1. Número de Formigas

Os gráficos 1 e 2 mostram o impacto do número de formigas na melhor solução e no tempo de execução. Foi determinado que os ganhos pelo aumento de população não justificam o aumento de tempo de execução. Por isso o número de formigas foi mantido em 10.

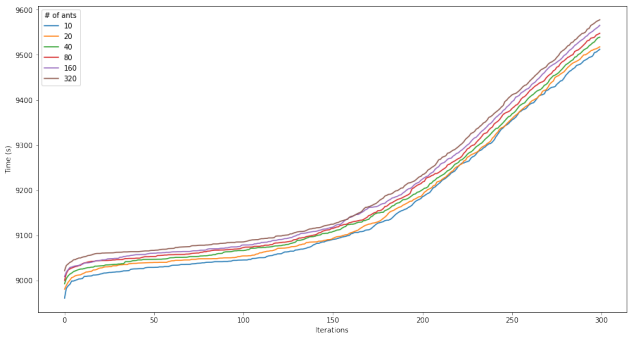

Figure 1. Melhor solução encontrada para diversos números de formigas

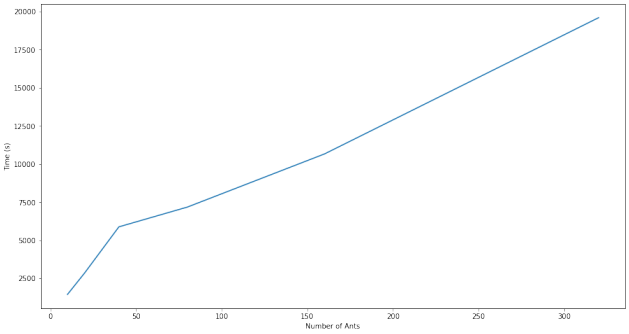

Figure 2. Tempo de execução por número de formigas

3.2. Iterações

O gráfico 3 mostra a melhor solução encontrada por iteração, e o gráfico 4 mostra o tempo de execução do teste para números máximos de iterações diferentes. Como o algoritmo é $O(n)$ em relação ao número de iterações e há ganhos até 700 iterações, foi escolhido 700 para o número de iterações.

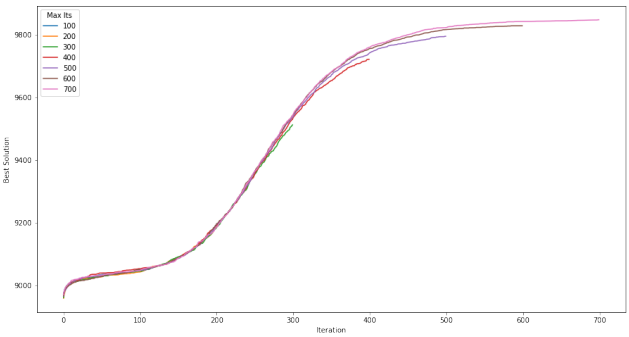

Figure 3. Melhor solução encontrada por iteração

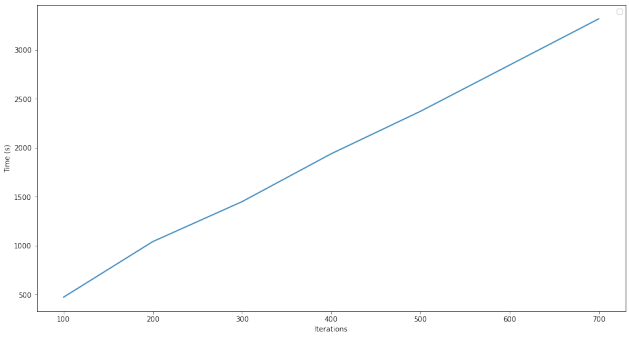

Figure 4. Tempo de execução por iteração

3.3. Beta

O gráfico 5 mostra o valor da melhor solução encontrada para diferentes valores de β. O β basicamente é o peso que

damos para seguir a aresta de maior comprimento, e como quanto maior o β melhor a solução, surge uma dúvida se na verdade a heurística de vizinhos mais distantes é melhor que o ACO implementado. Para testar essa hipótese foi encontrada a melhor solução para essa heurística setando α em 0 e iniciando uma formiga em cada vértice do grafo. Ficou claro que os feromônios ainda tinham um papel importante, e a implementação do ACO foi justificada. Foi escolhido $\beta = 75$.

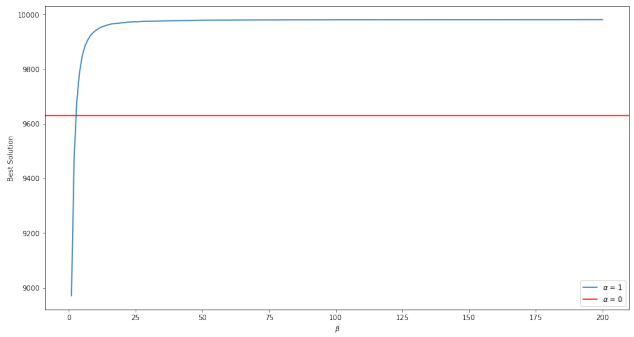

Figure 5. Melhor solução encontrada para diferentes valores de β

3.4. Rho

O gráfico 6 mostra o valor da melhor solução para diferentes valores de ρ, como 0.08 obteve os melhores resultados e uma convergência mais suave, optou-se por utilizar $\rho = 0.08$.

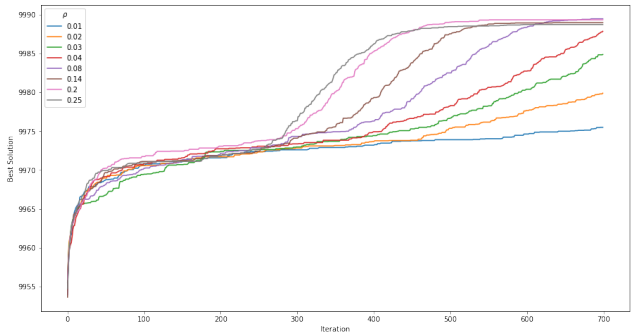

Figure 6. Melhor solução encontrada para diferentes valores de ρ

3.5. Alpha

Talvez não faça tanto sentido ajustar α, uma vez que o ajuste de β já seja um ajuste indireto de α, porém com objetivo explorativo, também realizamos um experimento para α. Os resultados estão ilustrados pelo gráfico 7. Foi observado que $\alpha = 2$ obteve resultados marginalmente melhores que $\alpha = 1$ consistentemente em múltiplas execuções do teste. Isso provavelmente ocorreu pois o ajuste de ρ alterou o impacto que os feromônios têm para encontrar um melhor caminho. Optou-se então por utilizar $\alpha = 2$.

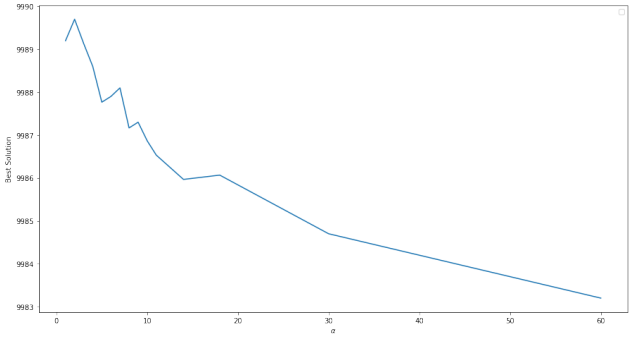

Figure 7. Melhor solução encontrada para diferentes valores de α

3.6. Parâmetros Finais

A tabela 2 apresenta os parâmetros escolhidos. E o gráfico 8 apresenta o resultado da execução com esses parâmetros.

Table 2. Parâmetros finais

Parâmetro	Valor
Iterações	700
Número de Formigas	10
Beta	75
Rho	0.08
Alpha	2

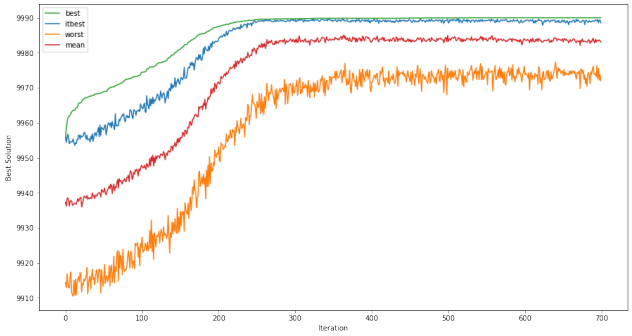

Figure 8. Melhor solução até então (best-so-far), melhor solução da iteração atual, pior solução e solução média para os parâmetros finais.

Na sessão seguinte iremos descrever os resultados do algoritmo para os três conjuntos de dados.

4. Resultados

Nessa sessão iremos apresentar três tabelas (3, 4, 5) com estatísticas descritivas sobre as soluções encontradas.

Table 3. Resultados para *entrada1.txt*

count	mean	std
30	990	0

Table 4. Resultados para *entrada2.txt*

count	mean	std	min	25%	max
30	175.56667	0.93526	173	176	176

Table 5. Resultados para *entrada3.txt*

count	mean	std
30	9990	0

5. Conclusão

O fato do desvio padrão para as entradas 1 e 3 serem 0 é um indicio de que possivelmente o algoritmo encontrou o ótimo toda vez. Para a entrada 2 temos que no percentil $.25$ encontramos o ótimo. Isso também indica que para os parâmetros finais o número de formigas e o número de iterações são maiores do que o necessário, isso é evidenciado ainda mais pelo gráfico 8 que na iteração 300 o algoritmo já havia convergido. Portanto poderíamos executar o algoritmo mais rapidamente sem comprometer qualidade da solução, por exemplo definindo uma condição de parada após x execuções sem melhora, ou simplesmente diminuindo o número máximo de execuções diretamente.

Seria interessante futuramente revisitar a etapa de otimização de parâmetros partindo dos parâmetros atuais, para otimizar o tempo de execução e qualidade das soluções. Seria interessante também alterar a ordem de otimização de parâmetros, por exemplo otimizando ρ antes de β.

References

[1] Marco Dorigo and Thomas Stutzle. *Ant colony optimization*. en. A Bradford Book. Cambridge, MA: Bradford Books, June 2004.