Frozen Lake Problem*

João E. Moreira[1][2020230563] joaomoreira@student.dei.uc.pt

[1] Universidade de Coimbra
[2] Departamento de Engenharia Informática

Abstract. Este trabalho investiga o desempenho de três algoritmos de procura utilizados para resolver o *Frozen Lake Problem*. As estratégias desenvolvidos foram: algoritmo evolucionário simples, evolução de *Qtables* e um modelo baseado em formigas. Cada algoritmo é avaliado em três mapas de diferentes tamanhos: 4×4, 8×8 e 12×12. Os resultados são apresentados em termos de eficácia na resolução do problema, qualidade das soluções encontradas, diversidade da população e capacidade de generalização para resolver novos mapas. Além disso, um desafio final é proposto para testar a robustez dos algoritmos em um mapa com características mais desafiadoras. Os resultados sugerem que o algoritmo baseado em formigas apresenta uma boa capacidade de generalização e robustez em comparação com os outros dois algoritmos testados, especialmente em cenários mais complexos.

Keywords: Meta-Heuristic · Optimization · Local Search · Global Search · Evolutionary Computation · Population-based algorithms · Ant Behavior Modeling · Ant Colony Optimization · Evolutionary Qtables .

1 Introdução

O problema do *Frozen Lake*, um dos desafios clássicos oferecidos pela biblioteca *Gymnasium*, apresenta um cenário onde um agente deve navegar por um terreno gelado, com o objetivo de alcançar um destino específico sem cair em lagos. A complexidade deste problema reside na incerteza da ausência de perceção do agente, o que torna a impossibilidade de usar algoritmos de aprendizagem por reforço.

Neste relatório, proponho a aplicação de técnicas de algoritmia evolucionária para resolver o problema do *Frozen Lake*. Os algoritmos evolucionários, inspirados nos processos biológicos de seleção natural e evolução, têm sido amplamente utilizados para resolver problemas de otimização, especialmente em espaços de procura complexos e vastos.

Primeiramente, farei algumas definições e notações. Em seguida, contextualizarei o problema do *Frozen Lake* e uma nova formulação deste problema. Posteriormente, apresentarei os fundamentos teóricos por trás das estratégias evolucionárias implementadas, com destaque para os princípios básicos e estratégias-chave, incluindo a representação do indivíduo, a função de *fitness* e os parâmetros do algoritmo.

* Supported by Nuno Lourenço and João Correia.

Por fim, apresentarei os resultados das experiências e uma comparação com outras abordagens mais simples e tradicionais. Além disso, analisarei as vantagens e limitações de nossa abordagem, bem como possíveis direções futuras para aprimoramento e expansão deste trabalho.

Em suma, este relatório visa explorar a eficácia e a aplicabilidade dos algoritmos evolucionários na resolução de problemas de navegação em ambientes complexos e incertos, utilizando o desafio do *Frozen Lake* como estudo de caso.

2 Definições e notações

Soluções admissíveis Uma solução admissível é qualquer solução que não viole as regras do problema. Neste relatório, considerei soluções admissíveis, aquelas que alcançam a saída sem violar nenhuma regra. Soluções que encontrem ótimos locais não são consideradas admissíveis.

Componentes O termo "componente" é usado ao longo deste relatório durante a descrição da construção das soluções. Nesse contexto, uma componente refere-se a um gene ou alelo e uma solução é uma coleção ou sequência de componentes.

3 Frozen Lake

O problema do *Frozen Lake* original, pode ser consultado em `https://gymnasium.farama.org/environments/toy_text/frozen_lake/` é ligeiramente diferente do que se apresenta nesta secção.

O agente começa inicia sempre no canto superior esquerdo e tem como objetivo, chegar ao canto inferior direito sem cruzar qualquer lago e vendado, ou seja, não sabe se o próximo passo dado cairá num lago ou continua a sua jornada. Os movimentos permitidos ao agente são:

- 0: Esquerda
- 1: Baixo
- 2: Direita
- 3: Cima

O ambiente é uma matriz quadrada de tamanho $n \times n$, com um número aleatório de lagos, distribuído de forma aleatória pelo terreno. Na tabela 1 está representada o tamanho dos indivíduos respetivamente às dimensões dos mapas que proponho-me a resolver.

4 Algoritmo Evolucionário Simples

Inicialmente abordei o problema com o algoritmo evolucionário simples de forma a compreender melhor o problema, para construir algoritmos que fossem mais capazes. Acabei por construir dois algoritmos com representações diferentes e que permitem utilizar operadores diferentes.

Dimensão	Tamanho máximo do individuo
4×4	100
8×8	200
12×12	500

Table 1. Correspondência do tamanho máximo dos indivíduos com as dimensões do mapa

4.1 Representação Inteira

Nesta representação, o genótipo dos indivíduos é uma string dos movimentos que pode fazer: $0, 1, 2, 3$. O objetivo é evoluir uma *policy* que o individuo deve seguir passo a passo.

Inicialização do Individuo Seguindo o princípio teórico de tentar cobrir o máximo possível do espaço de busca, o indivíduo é inicializado de forma construtiva, adicionando componentes aleatórias. Começamos com a solução vazia, $s = \{\}$, e adicionamos n componentes aleatórias à solução, onde o valor de n é escolhido aleatoriamente dentro do intervalo $[1, max\_individual\_size]$.

Isto resultará em indivíduos aleatórios, de tamanho dinâmico e aleatório.

Função de avaliação Para avaliar os melhores indivíduos criei uma função de avaliação que recompense pela proximidade à saída, por encontrar o caminho para a saída, e que penalizasse pela distância que ficou da saída e pelo tamanho do genótipo

$$\begin{cases} |x+y| \times 2 - |x-D| - |y-D| - \text{individual\_size} + 40 & \text{se admissível} \\ |x+y| \times 2 - |x-D| - |y-D| - \text{individual\_size} & \text{se não admissível} \end{cases} \tag{1}$$

sendo x e y a última coordenada na matriz onde o individuo ficou.

Seleção de Sobreviventes A seleção de sobreviventes não é um parâmetro deste algoritmo. No entanto, o melhor indivíduo da geração anterior é sempre incluído na nova geração para garantir que não se perde a melhor solução ao longo das gerações.

Seleção de Pais Utilizei seleção por torneio, onde k indivíduos são selecionados aleatoriamente e os dois melhores são usados no cruzamento de soluções.

Crossover Implementei o *two point crossover* para realizar a troca de informações entre duas soluções. Os pontos de corte são escolhidos aleatoriamente dentro do intervalo entre a primeiro e a última componente do indivíduo mais pequeno. Isso permite a troca de uma porção de informação entre os indivíduos sem modificar excessivamente a solução.

Mutação Foi implementado uma mutação uniforme, onde todas as componentes têm uma probabilidade de ser alteradas por outra componente. Também foi adicionada uma probabilidade de adicionar componentes ao final do genótipo para permitir que o indivíduo explore mais o espaço de procura e eventualmente saia de ótimos locais.

4.2 Representação em QTables

A ideia inicial era evoluir *Q-Tables*, visto que se trata de um problema muito aplicado à aprendizagem por reforço. No entanto, acabei por evoluir a *policy* a ser seguida para um determinado estado.

Sendo assim, o meu individuo é uma tabela das mesmas dimensões do mapa e em cada célula, há uma ação a tomar: $0, 1, 2, 3$:

Inicialização do Individuo Para cada posição na *Q-Table*, escolhemos uma ação aleatória. No entanto, ao gerar as componentes, não permito que os indivíduos colidam com as paredes. Por exemplo, se o agente estiver na primeira linha, não pode escolher aleatoriamente a componente que o leva para cima, apenas uma das outras três componentes restantes.

Função de Avaliação Para avaliar os indivíduos, precisamos executá-los na *Q-table*. No entanto, é possível que o agente fique preso em um ciclo infinito se a ação escolhida o fizer voltar a uma célula anteriormente visitada. Nesse caso, o episódio termina quando o agente retorna a uma célula já visitada.

A função de avaliação é a mesma da representação anterior.

$$\begin{cases} |x+y| \times 2 - |x-D| - |y-D| - \text{individual\_size} + 40 & \text{se admissível} \\ |x+y| \times 2 - |x-D| - |y-D| - \text{individual\_size} & \text{se não admissível} \end{cases} \tag{2}$$

sendo x e y a última coordenada na matriz onde o individuo ficou.

Seleção de Sobreviventes Utilizei a mesma estratégia de seleção por torneio da representação anterior.

Crossover Para compartilhar informações entre os pais, implementei o *crossover uniforme*, onde cada célula tem a mesma probabilidade de ser herdada do pai ou da mãe.

Mutação Para perturbar a solução, implementei a *mutação uniforme*, onde cada célula tem uma probabilidade de ser alterada para qualquer uma das outras ações. Neste caso, o agente pode inadvertidamente colidir com as paredes.

5 Formigas

Como sabemos, as formigas são excelentes abordagens para encontrar caminhos mais curtos. No entanto, no nosso problema não temos garantias se a formiga chega ao final, pois é mais provável que já tenha caído num buraco antes disso. Sendo assim, isto levanta-nos a dificuldade de como depositamos feromonas.

5.1 Representação

A representação das soluções segue o mesmo principio da primeira solução proposta, ou seja, uma *policy* que o agente deve seguir para chegar até ao fim ou cair num lago.

Geração de individuos A formiga adiciona uma componente à solução vazia de acordo com a quantidade de feromonas presente naquela aresta do grafo. A escolha da componente continua a ser estocástica, mas está dependente da quantidade e da qualidade das formigas que ali passaram.

Função de Avaliação Aqui é onde a minha abordagem diverge de muitos dos algoritmos conhecidos pela literatura. Eu avalio as formigas de forma a maximizar a sua *fitness* e não a minimizar o valor *objetivo*, porque a natureza do problema assim o obriga.

A função usada recompensa a proximidade e por encontrar um caminho para a saída e penaliza o tamanho do individuo

$$|x + y|/individual\_size + 10 \tag{3}$$

Na literatura é comum ter a função objetivo

$$1/individual\_size$$

para penalizar as formigas menos eficientes e vice-versa. No caso deste problema, não posso aplicar isso, pois podemos cair num lago antes de chegar à saída.

Evapuração de Feromonas A formula de aplicação para a evaporação usada é dada por

$$\tau_j = \rho \times \tau_j, \quad \forall \tau_j \in \Im \tag{4}$$

Deposição de Feromonas Para depositar feromonas, todas as formigas depositam a quantidade equivalente ao seu valor de *fitness* por onde passaram

$$\tau_j = \tau_j + Fitness(A_k)^{\alpha} \tag{5}$$

Probabilidade de escolher uma componente De forma a calcular a qualidade de uma componente, aplicamos a seguinte formula

$$Pr(C_j)) = \frac{\tau_j}{\sum_{C_k \in N_i} \tau_k} \tag{6}$$

6 Setup Experimental

Nesta secção irei discutir todo o processo de *grid search* dos melhores parâmetros para os algoritmos construídos de forma a produzirem a melhor eficácia, a maior diversidade e melhor qualidade da solução final.

Na tabela 2, estão presentes as variáveis que precisamos de medir para tirar as conclusões sob os objetivos definidos.

Parâmetros	Valor
Fitness	Float
Encontrar a saída	Bool
Diversidade	Float

Table 2. Variáveis dependentes e os valores que podem tomar

Irei executar cada algoritmo para todos os conjuntos de parâmetros presentes nas tabelas 3 e 4, trinta vezes de forma a garantir que os resultados não estão dependentes da *seed* do gerador de números aleatórios. Estas trinta execuções terão uma *seed* fixada, para garantir as mesmas condições entre algoritmos e conjuntos de parâmetros diferentes.

No fim de cada uma das trinta iterações irei guardar o valor da *fitness* do melhor individuo, se o algoritmo encontrou um caminho para a saída, a diversidade da população e o melhor individuo.

Visto que os algoritmos têm funções de *fitness* diferentes é impossivel comparar esses valores. Por essa razão, no momento de avaliar o melhor individuo defini uma função de avaliação nova

$$\begin{cases} \frac{\text{max\_individual\_size} \times |x+y|}{\text{individual\_size}} & \text{se admissível} \\ \frac{|x+y|}{\text{individual\_size}} & \text{se não admissível} \end{cases} \tag{7}$$

onde o *max_individual_size* é o tamanho máximo permitido do individuo para cada mapa.

Para calcular a diversidade da população final, calculei a distância de *Hamming* para todos os pares de indivíduos e no fim calculei a média.

Na tabela 5 estão presentes as avaliações realizadas para cada algoritmo. À primeira vista podemos pensar que o algoritmo das formigas estão em desvantagem, e é verdade, mas é o suficiente para elas atingirem resultados excelentes.

Parâmetros	Níveis
Pool de pais	[0.05 0.1 0.15]
Crossover	[0.7 0.8 0.9]
Mutação	[0.01 0.05 0.1 0.15]

Table 3. Parâmetros para o SEA inteiro e Qtable

Parâmetros	Níveis
Alpha	[0.8 0.9 1]
Evaporação	[0.8 0.9 1]

Table 4. Parâmetros para as formigas

Algoritmo	Gerações	Tamanho da População
SEA	1000	100
Qtable	1000	100
Formigas	100	10

Table 5. Número de avaliações por algoritmo

6.1 Escolha de Testes Estatísticos

Distribuição dos dados Em ordem a testar se os dados seguem uma distribuição normal, aplicarei o *Shapiro–Wilk test* a cada conjunto de experiências.

Teste de hipóteses Como veremos na secção dos resultados, os nossos dados não seguem uma distribuição normal e são não paramétricos. Por essa razão usarei *Wilcoxon test* para comparar e perceber se existe diferença entre duas experiências, com correção de *Bonferroni*. Este teste só será usado para a *fitness* e a diversidade. Para o estudo da eficácia será usado um teste de proporções.

Critério de decisão Quando encontro duas experiências que estatisticamente são significativamente diferentes, irei dizer que a melhor configuração é aquela que tem maior média.

Todos os testes serão realizados para um grão de confiança de 95% e um valor de p de 0.05.

7 Resultados

Nesta secção, irei apresentar e discutir os resultados obtidos. Devido à grande quantidade de experiências realizadas e à extensão dos dados, os resultados completos serão disponibilizados em anexo, enquanto aqui destacarei os resultados mais relevantes.

É importante ressaltar que não foi realizada uma comparação estatística entre os algoritmos. O foco principal foi encontrar as melhores configurações para cada algoritmo. No entanto, farei uma análise analítica dos resultados obtidos pelos três algoritmos.

7.1 Normalização dos dados

Ao executar o teste de *Shapiro-Wilk*, foi observado que as amostras não seguem uma distribuição normal, pois houve evidências para rejeitar a hipótese nula.

Isso significa que o para os testes estatísticos devemos usar o teste de *Wilcoxon test* com correção de *Bonferroni* pelas razões apresentadas anteriormente.

7.2 Mapa 4x4

Nas Tabelas 6 e 7 estão presentes os resultados das execuções para o mapa de dimensão 4×4.

Repara que segundo a natureza deste mapa de dimensão 4 em que a solução ótima global toma 8 componentes, esta solução terá uma qualidade $\frac{100 \times |4+4|}{8} = 100$, ou seja, para estes casos a qualidade é igual ao tamanho máximo possível para o individuo.

Todos os algoritmos conseguiram encontrar um caminho admissível para este mapa. No entanto, o *SEA*, mesmo com a melhor configuração encontrada, gerou soluções que não foram capazes de alcançar um caminho ótimo global.

Em relação à diversidade, observamos que o algoritmo de *Formigas* converge completamente para a solução ótima, enquanto o algoritmo de *Qtables* apresenta uma grande diversidade na população. Isso poderá ocorrer porque os indivíduos são tabelas, contendo mais componentes, o que aumenta a probabilidade de diferenças em cada célula.

Algoritmo	Qualidade	Admissível	Diversidade
SEA	92.3 ± 15.0	1.0 ± 0.0	2.5 ± 1.63
Qtable	100.0 ± 0.0	1.0 ± 0.0	9.2 ± 0.98
Formigas	100.0 ± 0.0	1.0 ± 0.0	0.0 ± 0.0

Table 6. Média e desvio padrão da melhor configuração encontrada para o mapa 4×4

Para esta instância, há uma variedade de parâmetros que resultam em 100% de admissibilidade. No caso das *Formigas*, qualquer configuração de parâmetros resulta em 100% de admissibilidade. Para os outros algoritmos, sugiro consultar os resultados completos em anexo.

7.3 Mapa 8x8

Nas Tabelas 8 e 9 estão presentes os resultados das execuções para o mapa de dimensão 8×8.

Algoritmo	Dimensão Pool	Crossover	Mutação	Evaporação	Alpha
Qualidade					
SEA	0.05	0.8	0.15	-	-
Qtable	qualquer	qualquer	qualquer	-	-
Formigas	-	-	-	qualquer	qualquer
Diversidade					
SEA	0.1	0.7	0.15	-	-
Qtable	0.15	0.7	0.15	-	-
Formigas	-	-	-	qualquer	qualquer
Admissível					
SEA	0.05	0.7	0.01	-	-
Qtable	0.05	0.7	0.15	-	-
Formigas	-	-	-	qualquer	qualquer

Table 7. As melhores configurações para o mapa de 4x4

Nesta instância, a diversidade da colónia de formigas aumentou consideravelmente em comparação com o mapa anterior de 4×4. Todos os algoritmos alcançaram uma eficácia de 100%. No entanto, o *SEA* ainda não foi capaz de encontrar a solução ótima global.

Algoritmo	Qualidade	Admissível	Diversidade
SEA	170.2 ± 37.8	1.0 ± 0.0	4.3 ± 2.42
Qtable	200.0 ± 0.0	1.0 ± 0.0	$34.47.2 \pm 3.41$
Formigas	200.0 ± 0.0	1.0 ± 0.0	24.07 ± 8.61

Table 8. Média e desvio padrão da melhor configuração encontrada para o mapa 8×8

É compreensível que atingir o ótimo global nem sempre seja uma tarefa fácil, mas diante desta instância, os outros algoritmos mantiveram uma boa performance, ao passo que o *SEA* não consegue acompanhar.

Algo de interessante de ver aqui é que no algoritmo de *formigas* toda a variedade de parâmetros deixa de funcionar corretamente e temos de passar à escolha dos mesmo. Esta pode ser a cause para conseguirmos obter uma maior diversidade.

Novamente, observamos uma variedade de configurações de parâmetros que resultam em 100% de admissibilidade. Recomendo consultar os resultados completos em anexo para mais detalhes.

7.4 Mapa 12x12

Nas Tabelas 10 e 11 estão presentes os resultados das execuções para o mapa de dimensão 12×12.

Algoritmo	Dimensão Pool	Crossover	Mutação	Evaporação	Alpha
Qualidade					
SEA	0.05	0.9	0.15	-	-
Qtable	0.05	0.7	0.15	-	-
Formigas	-	-	-	qualquer	qualquer
Diversidade					
SEA	0.2	0.7	0.15	-	-
Qtable	0.1	0.7	0.05	-	-
Formigas	-	-	-	0.9	0.8
Admissível					
SEA	0.05	0.7	0.01	-	-
Qtable	0.05	0.7	0.1	-	-
Formigas	-	-	-	qualquer	qualquer

Table 9. As melhores configurações para o mapa de 8x8

Algoritmo	Qualidade	Admissível	Diversidade
SEA	416.5 ± 44.7	1.0 ± 0.0	4.3 ± 2.09
Qtable	500.0 ± 0.0	1.0 ± 0.0	58.57 ± 13.26
Formigas	500.0 ± 0.0	1.0 ± 0.0	73.4 ± 27.19

Table 10. Média e desvio padrão da melhor configuração encontrada para o mapa 12×12

É possível observar que os algoritmos de *Qtables* e *Formigas* mantêm-se altamente eficientes e produzem soluções de excelente qualidade. No entanto, para esta instância em particular, a colónia de formigas demonstra ter uma maior diversidade. Para entender essa observação, podemos examinar a Figura 1, que mostra a quantidade de feromonas depositados em cada aresta do mapa.

Atenção que o canto inferior esquerdo corresponde ao ponto de partida e o canto superior direito ao da saída no mapa.

Na Figura 1, observe que há uma variedade de caminhos possíveis que levam em direção à saída, o que sugere uma diversidade de soluções encontradas pela colónia de formigas. Suponha que, se houver formigas suficientes para explorar todo o espaço de busca, o comportamento do algoritmo de formigas poderá ser semelhante ao do *Qtable*. No entanto, como usamos dez vezes menos indivíduos no algoritmo de formigas, torna-se difícil fazer essa comparação direta.

Quanto às configurações dos algoritmos, os resultados mostram que não há uma configuração clara que se destaque como a melhor para todas as métricas consideradas. No entanto, podemos notar que o algoritmo de *Qtables* atinge resultados semelhantes com 100 vezes a mais avaliações do que o algoritmo de *Formigas*.

Em resumo, até o momento, é difícil determinar qual algoritmo é o melhor, pois ambos produzem resultados de alta qualidade. Na próxima secção, vamos

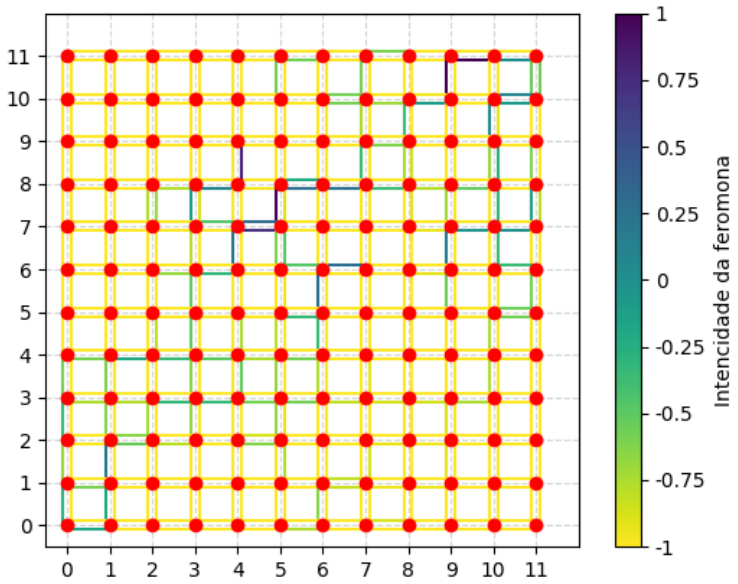

Fig. 1. Quantidade de feromonas depositas nas arestas

Algoritmo	Dimensão Pool	Crossover	Mutação	Evaporação	Alpha
Qualidade					
SEA	0.1	0.8	0.15	-	-
Qtable	0.15	0.7	0.15	-	-
Formigas	-	-	-	qualquer	qualquer
Diversidade					
SEA	0.1	0.7	0.15	-	-
Qtable	0.2	0.7	0.15	-	-
Formigas	-	-	-	0.8	0.9
Admissível					
SEA	0.05	0.8	0.1	-	-
Qtable	0.15	0.7	0.15	-	-
Formigas	-	-	-	0.9	0.8

Table 11. As melhores configurações para o mapa de 12x12

submeter ambos os algoritmos a um mapa de alta dificuldade e analisar os resultados, com o objetivo de uma distinção entre eles.

8 Desafio

Uma última avaliação que gostaria de realizar é testar a capacidade de generalização do algoritmo para resolver outros mapas para além dos propostos.

Para isso, irei fixar os parâmetros da Tabela 11 para os algoritmos de *Qtables* e de *Formigas*, e então verificar se os algoritmos continuam a ser capazes de resolver um novo mapa em uma média de trinta execuções. Isto é uma simplificação, mas poderá ser o suficiente para tirar uma conclusão.

Fig. 2. Mapa de dimensão 12, seed 2

Note que o mapa presente na figura 2 tem uma dificuldade acrescida para a maioria dos mapas. Para além de conter células que facilmente se tornam em ótimos locais, o agente é obrigado a subir para resolver o mapa.

Algoritmo	Total de execuções	Total de execuções completas
Qtable	30	2
Formigas	30	21

Table 12. Total de execuções completas para o mapa 12×12 com a *seed* 2

Como podemos ver na Tabela 12, para este novo mapa, ambos os algoritmos tiveram uma queda na eficácia. No entanto, o algoritmo de formigas conseguiu resolver dois terços das execuções, o que demonstra maior robustez diante do algoritmo de *Qtables*.

Na Figura 2, é possível ver o resultado de uma das execuções do algoritmo de *Qtables*, onde a maioria das execuções tinha como destino a célula onde o agente se encontra inicialmente posicionado, um ótimo local.

9 Conclusão

Os resultados obtidos demonstram que os algoritmos de *Qtables* e de *Formigas* se destacam, tanto em termos de eficácia na resolução do problema quanto na qualidade das soluções encontradas.

Ao longo do estudo, explorei diferentes representações e operadores para o algoritmo evolucionário simples, para além de implementar uma versão dos algoritmos de colónias de formigas. Os resultados experimentais mostraram que esses algoritmos foram capazes de resolver o problema com sucesso, encontrando soluções admissíveis e minimizando a distância percorrida pelo agente.

Em suma, este estudo contribui para o entendimento e aplicação de algoritmos de busca em problemas práticos de navegação. Além disso, destaca a importância da escolha adequada dos parâmetros corretos para obter resultados eficazes e robustos. O trabalho abre oportunidades para pesquisas futuras, incluindo a alteração do tipo de mutação no algoritmo de *Qtables* de forma a que não permita o agente bater nas paredes.

References

1. Author, A.E. Eiben, Author, J.E. Smith.: Introduction to Evolutionary Computing.(2015)
2. Author, Marco Dorio, Author, Thomas Stützle.: Ant Colony Optimization. (2004)
3. Author, Sean Luke.: Essentials of Metaheuristics. 2nd edn. (2012)
4. Author, Rafael Martí, Author, panos Pardalos, Author, Mauricio Resend.: HandBook of Heuristics. (2018)