Modelagem Computacional e Análise Comparativa de Têmpera Simulada e Algoritmos Genéticos para o Problema do Caixeiro Viajante

Miguel Dos Anjos Brack[1], **Renzo Tognella De Rosa**[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR) - Campus Curitiba
Av. Sete de Setembro, 3165 - Rebouças, Curitiba - PR, 80230-901

miguelanjosbrack@alunos.utfpr.edu.br, renzotognella@alunos.utfpr.edu.br

Abstract. *Analyzes parameter impact on Simulated Annealing (SA) and Genetic Algorithms (GA) for the 500-city TSP. Found critical parameter sensitivity: SA yields top quality at extreme computational cost (requiring slow cooling), while GA offers better cost-benefit needing careful tuning. Highlights performance trade-offs between SA and GA.*

Resumo. *Analisa o impacto paramétrico em Têmpera Simulada (SA) e Algoritmos Genéticos (GA) para o TSP de 500 cidades. Encontrou-se sensibilidade paramétrica crítica: SA alcança qualidade superior a custo computacional extremo (exigindo resfriamento lento), enquanto GA oferece melhor custo-benefício, necessitando de ajuste fino. Destacam-se os trade-offs de desempenho entre SA e GA.*

Palavras chave: Problema do Caixeiro Viajante, Otimização Combinatória, Metaheurísticas, Têmpera Simulada, Algoritmos Genéticos, Modelagem Computacional.

1. Introdução

O Problema do Caixeiro Viajante (TSP) busca a rota de menor custo que visita n cidades uma única vez, retornando à origem. Formalmente, dada uma matriz de custos $C = [c_{ij}]$, busca-se a permutação $\pi = (\pi_1, \ldots, \pi_n)$ que minimiza:

$$\text{Custo Total}(\pi) = c_{\pi_n \pi_1} + \sum_{i=1}^{n-1} c_{\pi_i \pi_{i+1}} \tag{1}$$

Este problema é um arquétipo da classe NP-difícil [Garey and Johnson 1979], com um espaço de busca de tamanho $O(n!)$, tornando métodos exatos e buscas clássicas inviáveis para instâncias de tamanho relevante, como as encontradas em logística ou planejamento. Buscas cegas (BFS, DFS) explodem combinatoriamente. Estratégias gulosas (e.g., Vizinho Mais Próximo) são rápidas mas frequentemente resultam em soluções muito ruins, presas em decisões míopes. Buscas informadas como A* demandam recursos exponenciais de memória/tempo para n grande. Buscas locais simples (e.g., Hill Climbing com 2-opt) ficam estagnadas no primeiro ótimo local encontrado.

Diante disso, metaheurísticas como Têmpera Simulada (SA) [Kirkpatrick et al. 1983] e Algoritmos Genéticos (GA) [Goldberg 1989] surgem como alternativas pragmáticas. Este trabalho foca na aplicação de SA e GA a uma instância TSP de 500 cidades, com ênfase especial na **análise do impacto dos parâmetros de entrada** (T_0, α, T_f para SA; P, G, p_m para GA) no desempenho e no custo computacional. Comparamos suas performances entre si e justificamos suas vantagens sobre as buscas clássicas.

2. Metodologia

Esta seção detalha as escolhas metodológicas para a aplicação da Têmpera Simulada (SA) e dos Algoritmos Genéticos (GA) ao TSP, utilizando uma instância de 500 cidades.

2.1. Modelagem para Têmpera Simulada (SA)

- **Representação da Solução:** Uma permutação $\pi = (\pi_1, \ldots, \pi_n)$ dos índices das $n = 500$ cidades, representando a ordem de visita.
- **Função de Custo ($f(\pi)$):** O comprimento total da rota correspondente à permutação π, calculado usando a fórmula do calculo de custo com distâncias Euclidianas.
- **Geração de Vizinhança:** Utilizou-se o operador **2-opt**. Este operador seleciona duas arestas não adjacentes (v_i, v_{i+1}) e (v_j, v_{j+1}) na rota atual e as substitui por (v_i, v_j) e (v_{i+1}, v_{j+1}), invertendo o segmento da rota entre v_{i+1} e v_j. É um movimento clássico e eficaz para o TSP [Reinelt 1994].
- **Funcionamento do SA:**
  - *Inicialização:* A busca começa com uma rota inicial gerada aleatoriamente (permutação aleatória).
  - *Parâmetros Chave:* Temperatura inicial (T_{inicial}), temperatura final (T_{final}) e taxa de resfriamento (α). T_{inicial} deve ser alta o suficiente para permitir exploração inicial, T_{final} define o critério de parada, e α controla a velocidade do resfriamento [Russell and Norvig 2020].
  - *Iteração:* Em cada iteração, uma solução vizinha π' é gerada aplicando o operador 2-opt à solução atual π.
  - *Critério de Aceitação (Metropolis):* A mudança de π para π' é avaliada. Se $\Delta E = f(\pi') - f(\pi) < 0$ (melhora), a nova solução π' é sempre aceita. Se $\Delta E \geq 0$ (piora), π' é aceita com uma probabilidade $P = \exp(-\Delta E/T)$, onde T é a temperatura atual. Isso permite que o algoritmo escape de ótimos locais [Kirkpatrick et al. 1983].
  - *Esquema de Resfriamento:* Utilizou-se um resfriamento geométrico: $T_{\text{nova}} = T_{\text{atual}} \times \alpha$, onde $0 < \alpha < 1$. O valor de α foi variado nos experimentos. A busca termina quando $T \leq T_{\text{final}}$.

2.2. Modelagem para Algoritmos Genéticos (GA)

- **Representação do Cromossomo:** Assim como na SA, cada indivíduo (cromossomo) na população é representado por uma permutação $\pi = (\pi_1, \ldots, \pi_n)$ dos índices das cidades.
- **Função de Aptidão (Fitness):** Como o GA tipicamente maximiza a aptidão, utilizou-se o inverso do custo da rota: $\text{Fitness}(\pi) = 1/f(\pi)$. Rotas mais curtas têm maior aptidão.
- **Operadores Genéticos [Eiben and Smith 2015]:**
  - *Seleção:* Implementou-se a **Seleção por Torneio**, onde um pequeno número de indivíduos é selecionado aleatoriamente da população, e o melhor entre eles é escolhido para reprodução.
  - *Cruzamento (Crossover):* Utilizou-se o **Order Crossover (OX)**. Este operador preserva a ordem relativa e a posição de um subsegmento aleatório de um dos pais, preenchendo o restante do filho com cidades do outro pai na ordem em que aparecem, evitando duplicatas. É adequado para representações baseadas em permutação.
  - *Mutação:* Aplicou-se a **Mutação por Inversão**, que seleciona aleatoriamente um segmento da rota e inverte a ordem das cidades dentro desse segmento (similar ao 2-opt, mas aplicado com baixa probabilidade).

- **Funcionamento do GA:**
  - *Inicialização:* Uma população inicial de `tam_pop` indivíduos (rotas) é gerada aleatoriamente.
  - *Parâmetros Chave:* Tamanho da população (`tam_pop`), número de gerações (`geracoes`) e taxa de mutação (`taxa_mut`). Estes parâmetros foram variados experimentalmente [Russell and Norvig 2020].
  - *Ciclo Evolutivo:* A cada geração, a aptidão de cada indivíduo é avaliada. O **elitismo** foi empregado, garantindo que o(s) melhor(es) indivíduo(s) da geração atual passe(m) diretamente para a próxima. Em seguida, ocorrem os passos de seleção (Torneio), cruzamento (OX) e mutação (Inversão) para formar a nova população.
  - *Critério de Parada:* O algoritmo termina após um número pré-definido de `geracoes`. Múltiplas execuções com diferentes configurações de parâmetros foram realizadas.

3. Resultados e Análise Focada nos Parâmetros

Os experimentos variaram os parâmetros chave de SA e GA para a instância `grafo_500.csv`. As tabelas detalhadas estão disponíveis no repositório de código; focamos aqui na análise do impacto paramétrico.

3.1. Impacto dos Parâmetros na SA

A performance da SA mostrou-se **extremamente sensível** ao esquema de resfriamento:

- **Taxa de Resfriamento (α):** Este é o fator dominante. Valores de α muito próximos de 1 (e.g., 0.999999) foram *indispensáveis* para atingir a melhor qualidade (custo $\approx$ 17381). Isso implica um resfriamento extremamente lento, necessitando de um número massivo de iterações (10^9 a 10^{10}), evidenciando um custo computacional altíssimo para o refinamento final. Valores de α menores (e.g., ≤ 0.9999) levaram a uma convergência muito mais rápida, porém para soluções significativamente piores (custos > 20000), presas em ótimos locais.
- **Temperaturas ($T_{\text{inicial}}, T_{\text{final}}$):** A T_{inicial} precisou ser alta (e.g., $10^4 - 10^5$) para permitir que a busca escapasse de configurações iniciais ruins. T_{final} baixa (e.g., ≤ 0.001) foi necessária para a convergência final. O intervalo $T_{\text{inicial}}/T_{\text{final}}$ e α estão interligados, determinando o número total de iterações.

A SA pode atingir excelente qualidade, mas o custo para isso é fortemente dependente de um α que permita exploração prolongada.

3.2. Impacto dos Parâmetros no GA

O GA também é sensível aos parâmetros, mas talvez de forma mais multifacetada:

- **Taxa de Mutação (p_m):** Essencial para o equilíbrio exploração-explotação. Taxas muito baixas (< 0.01) levaram à perda de diversidade e convergência prematura para soluções medianas (custo > 25000). Taxas muito altas (> 0.2) introduziram ruído excessivo, tornando a busca quase aleatória e impedindo a convergência para boas soluções (custo > 20000). Uma faixa intermediária ($p_m \approx 0.015 - 0.15$) produziu os melhores resultados (custo $\approx$ 18160 - 19400).
- **Tamanho da População (P):** Populações maiores (e.g., $P = 100, 200$) geralmente superaram populações menores ($P = 10, 25, 50$), pois mantêm maior diversidade genética, reduzindo o risco de estagnação. O benefício, contudo, vem ao custo de mais avaliações por geração.

- **Número de Gerações (G):** Mais gerações (3×10^5 a 10^6) permitiram maior refinamento, mas com retornos decrescentes. O custo total (avaliações $\approx P \times G$) para as melhores soluções do GA ($10^7 - 10^8$) foi significativamente menor que o da SA para sua melhor solução.

O GA exige um ajuste cuidadoso, especialmente de p_m, para balancear a busca, oferecendo um bom compromisso qualidade/custo.

3.3. Comparativo SA vs. GA e Vantagens sobre Buscas Clássicas

SA vs. GA: A SA demonstrou potencial para maior qualidade final (custo 17381 vs 18160 do GA), mas com um custo computacional ordens de magnitude maior e dependência crítica de um resfriamento muito lento (α). O GA apresentou um melhor custo-benefício geral, atingindo soluções muito boas com esforço computacional consideravelmente menor, embora exija atenção ao balanceamento de seus múltiplos parâmetros (P, G, p_m).

Justificativa vs. Clássicos: A inadequação das buscas clássicas para o TSP de 500 cidades reside em:

- *Complexidade Bruta:* Buscas cegas (BFS/DFS) são impossíveis pelo espaço $O(n!)$.
- *Ótimos Locais:* O espaço de busca do TSP é repleto de ótimos locais. Métodos gulosos (NN) e buscas locais simples (Hill Climbing com 2-opt) ficam presos no primeiro que encontram, sem capacidade de explorar além.
- *Recursos:* A* requer memória/tempo exponencial no pior caso, inviável para $n = 500$.

As metaheurísticas superam essas limitações:

- **SA:** Seu mecanismo de aceitação probabilística ($P = \exp(-\Delta E/T)$) permite explicitamente movimentos que pioram a solução (especialmente com T alto), possibilitando *escapar de vales de ótimos locais* e explorar outras regiões do espaço de busca.
- **GA:** A manutenção de uma *população diversa* e o uso de *recombinação (crossover)* permitem explorar múltiplas regiões do espaço em paralelo e combinar características de boas soluções. A *mutação* reintroduz diversidade, prevenindo a estagnação de toda a população em um único ótimo local.

Esses mecanismos conferem a SA e GA a robustez necessária para navegar eficientemente na complexa paisagem de busca do TSP.

3.4. Comparativo SA vs. GA e Vantagens sobre Buscas Clássicas

SA vs. GA: A SA demonstrou potencial para maior qualidade final (custo 17381 vs 18160 do GA), mas com um custo computacional ordens de magnitude maior e dependência crítica de um resfriamento muito lento (α). O GA apresentou um melhor custo-benefício geral, atingindo soluções muito boas com esforço computacional consideravelmente menor, embora exija atenção ao balanceamento de seus múltiplos parâmetros (P, G, p_m).

Justificativa vs. Clássicos: A inadequação das buscas clássicas para o TSP de 500 cidades reside em:

- *Complexidade Bruta:* Buscas cegas (BFS/DFS) são impossíveis pelo espaço $O(n!)$.
- *Ótimos Locais:* O espaço de busca do TSP é repleto de ótimos locais. Métodos gulosos (NN) e buscas locais simples (Hill Climbing com 2-opt) ficam presos no primeiro que encontram, sem capacidade de explorar além.
- *Recursos:* A* requer memória/tempo exponencial no pior caso, inviável para $n = 500$.

As metaheurísticas superam essas limitações:

- **SA:** Seu mecanismo de aceitação probabilística ($P = \exp(-\Delta E/T)$) permite explicitamente movimentos que pioram a solução (especialmente com T alto), possibilitando *escapar de vales de ótimos locais* e explorar outras regiões do espaço de busca.
- **GA:** A manutenção de uma *população diversa* e o uso de *recombinação (crossover)* permitem explorar múltiplas regiões do espaço em paralelo e combinar características de boas soluções. A *mutação* reintroduz diversidade, prevenindo a estagnação de toda a população em um único ótimo local.

Esses mecanismos conferem a SA e GA a robustez necessária para navegar eficientemente na complexa paisagem de busca do TSP.

4. Conclusão

Avaliamos SA e GA para o TSP (500 cidades) usando 2-opt (SA), seleção por torneio, OX e Inversão (GA). Ambas encontraram soluções de alta qualidade ($\approx$ 17.400 SA, $\approx$ 18.200 GA), mas dependentes de parâmetros. SA obteve o melhor custo, porém com esforço computacional massivo ($\approx 10^{10}$ iterações). GA foi mais eficiente ($\approx 10^{8}$ avaliações), embora sensível à parametrização. Ambas superam buscas clássicas; GA apresentou melhor custo-benefício, SA maior capacidade de refinamento. O estudo reforça a eficácia das metaheurísticas e a importância da modelagem e sintonia de parâmetros.

5. Contribuição dos Autores e Código Fonte

- Renzo Tognella De Rosa: Otimização dos algoritmos de Têmpera Simulada e Algoritmo Genético, coleta de resultados e organização dos dados coletados.
- Miguel Dos Anjos Brack: Implementação inicial do algoritmo de Têmpera Simulada, adaptação da estrutura para leitura de arquivos CSV e visualização SVG, refatoração do Algoritmo Genético, criação dos gráficos de custo e temperatura, e organização geral do projeto.

O código fonte das implementações está disponível em: `https://github.com/Miguel3074/Trabalho_Si`

Referências

[Eiben and Smith 2015] Eiben, A. E. and Smith, J. E. (2015). *Introduction to Evolutionary Computing*. Springer, 2nd edition.

[Garey and Johnson 1979] Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. WH Freeman.

[Goldberg 1989] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional.

[Kirkpatrick et al. 1983] Kirkpatrick, S., Gelatt Jr, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

[Reinelt 1994] Reinelt, G. (1994). *The Traveling Salesman: Computational Solutions for TSP Applications*, volume 840. Springer Science & Business Media.

[Russell and Norvig 2020] Russell, S. J. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition.