Universidade Federal do ABC
Centro de Matemática, Computação e Cognição
Projeto de pesquisa para o programa IC – Edital 04/2022

Algoritmos de Aproximação para o $(1, 2)$-TSP

Junho/2022

Resumo

No problema do Caixeiro Viajante temos um conjunto de cidades juntamente com os custos de viajar de uma cidade para outra e queremos encontrar uma rota de custo mínimo que passa por todas elas exatamente uma vez. Esse é um problema clássico e central em otimização combinatória, com diversas aplicações práticas e, infelizmente, NP-difícil. Este projeto tem por objetivo a introdução do candidato à pesquisa científica por meio do estudo de diferentes algoritmos para o problema mencionado, com foco em algoritmos de aproximação. Cada algoritmo envolve técnicas e conceitos diferentes, possibilitando assim ao aluno complementar sua formação em Ciência da Computação.

1 Introdução

Problemas em Otimização Combinatória têm como objetivo encontrar a melhor solução dentro de um enorme mas finito conjunto de soluções possíveis. Em geral, um problema desses possui um conjunto de restrições que define o que é uma solução viável e uma função objetivo que determina o valor de cada solução viável. Um exemplo é o clássico problema do Caixeiro Viajante (TSP, de *Traveling Salesman Problem*), em que temos um conjunto de cidades e os custos entre pares delas que representam, por exemplo, a distância entre duas cidades. O objetivo é encontrar um ciclo que passe por todas as cidades exatamente uma vez (Hamiltoniano) e que tenha custo mínimo.

O TSP aparece em diferentes contextos ao longo da história e essa característica contribui para sua importância no estudo de algoritmos. Relatos e documentos que datam do século 19 mostram tentativas de mercadores europeus de criarem mapas com rotas mercantes entre determinadas cidades, a fim de otimizar o percurso no quesito tempo ou dificuldade do trajeto [2]. Como exemplo temos a campanha publicitária da empresa Procter & Gamble do ano 1960, onde era oferecido um prêmio de $10,000 à pessoa que apresentasse o menor percurso entre determinadas cidades dos Estados Unidos. O TSP esteve presente em jogos também, como o *The Traveller's Dodecahedron: A Voyage Round the World*. Esse jogo tinha a proposta de criar um ciclo ao longo das arestas de um dodecaedro permitindo, no entanto, a repetição de arestas para tal [9].

Também é possível perceber aplicações do TSP nos dias atuais. Por exemplo, quando se quer projetar a melhor rota de uma linha de ônibus dentro de uma cidade. Ou quando se deseja visitar os principais pontos turísticos em uma viagem, planejando a melhor rota que permitiria tal visita em tempo hábil. Ou até mesmo otimizar a quantidade "giros" que telescópios de larga escala devem fazer para observar um determinado número de galáxias e planetas [2]. Evidencia-se, portanto, a importância e relevância que o TSP possui.

Apesar de bem relacionado aos problemas de caminho mínimo e árvore geradora mínima, ambos resolvíveis em tempo polinomial, o TSP é NP-difícil [6], o que significa que, a menos

que $\mathsf{P} = \mathsf{NP}$, não se tem esperança de criar um algoritmo que sempre devolva uma solução ótima em tempo polinomial para qualquer entrada. No entanto, dada a importância desse problema, ainda assim é desejável encontrar boas soluções para o mesmo. Uma abordagem clássica para lidar com problemas NP-difíceis é a de algoritmos de aproximação, que são algoritmos que executam em tempo polinomial sobre qualquer entrada e garantem que a solução devolvida tenha um custo que está a no máximo um certo fator do custo de uma solução ótima.

Infelizmente, para o TSP, não é possível encontrar algoritmos de aproximação para grafos em geral [11], a menos que $\mathsf{P} = \mathsf{NP}$. Uma forma de contornar isso é lidar com instâncias específicas, como aquelas em que os custos satisfazem a desigualdade triangular ou aquelas em que os custos são apenas 1 ou 2. O problema (1,2)-TSP considera apenas instâncias que satisfazem este último caso.

Este projeto tem por objetivo a introdução do candidato à pesquisa científica, bem como a complementação de sua formação em Ciência da Computação, iniciando seu conhecimento na área de otimização combinatória por meio do estudo de algoritmos de aproximação para o (1,2)-TSP. O restante desse documento está dividido da seguinte forma. Na Seção 2 apresentamos alguns conceitos que são relevantes para este projeto de pesquisa. Na Seção 3 apresentamos formalmente os problemas de interesse bem como listamos alguns resultados existentes sobre eles. Na Seção 4 estabelecemos os objetivos deste projeto de pesquisa. Na Seção 5 falamos sobre a metodologia e a viabilidade deste projeto. Por fim, na Seção 6 indicamos o plano de trabalho e um possível cronograma de execução.

2 Conceitos básicos

O problema do Caixeiro Viajante é um problema de otimização combinatória que envolve grafos. Por isso, antes de defini-lo formalmente, precisamos de alguns conceitos nessas duas áreas, que são dados nas seções a seguir.

2.1 Otimização combinatória e algoritmos de aproximação

Problemas em Otimização Combinatória surgem naturalmente de aplicações práticas, como minimizar rotas de veículos, maximizar lucros, minimizar desperdício de material de produção, minimizar uso de recursos disponíveis, entre tantos outros. De modo geral, testar todos os elementos dentre as soluções possíveis na busca pela melhor mostra-se inviável na prática, mesmo para instâncias de tamanho moderado. Além disso, a maioria dos problemas interessantes são NP-difíceis, de forma que, a menos que $\mathsf{P} = \mathsf{NP}$, não existem (i) algoritmos eficientes para resolver estes problemas de (ii) forma ótima (iii) para qualquer instância. As diferentes abordagens que temos na literatura contornam essa dificuldade abrindo mão de

uma ou mais dessas três condições. Uma dessas envolve a criação de algoritmos de aproximação.

Intuitivamente, dizemos que um algoritmo é de aproximação se, para cada instância de um determinado problema, ele devolve uma solução cujo valor está garantidamente a um fator de distância do valor de uma solução ótima para aquela instância.

Seja $\mathcal{I}$ o conjunto de todas as entradas válidas para um determinado problema e $\mathcal{A}$ um algoritmo de tempo polinomial que devolve soluções viáveis para o mesmo. Para $I \in \mathcal{I}$, denotamos por $\mathcal{A}(I)$ o valor da solução criada por $\mathcal{A}$ para I e por $OPT(I)$ o menor valor de uma solução para I.

Dizemos que $\mathcal{A}$ é um **algoritmo de aproximação** para o problema em consideração se existe um valor α tal que para qualquer $I \in \mathcal{I}$ temos:

- $\mathcal{A}(I) \leq \alpha OPT(I)$ se o problema é de minimização; e
- $\mathcal{A}(I) \geq \frac{1}{\alpha} OPT(I)$ se o problema é de maximização.

Chamamos α de **fator de aproximação**. Também chamamos $\mathcal{A}$ de **α-aproximação**.

Pela definição, um fator de aproximação não precisa necessariamente ser um valor constante, apesar de que isso em geral é o desejável. Particularmente, note que quanto menor o valor de α, mais próximo de uma solução ótima estamos. Para alguns problemas, é possível obter uma família de algoritmos de aproximação com fatores bem próximos ao ótimo, denominada **esquema de aproximação de tempo polinomial** (PTAS, de *polynomial-time approximation scheme*). Um algoritmo de aproximação é um PTAS se para todo $\varepsilon > 0$ ele é uma $(1 + \varepsilon)$-aproximação.

2.2 Grafos

Um **grafo (simples)** é definido por um par (V, E), onde V é um conjunto finito de elementos, chamados **vértices**, e E é um conjunto de pares não ordenados dos elementos de V, chamados **arestas**. Utilizaremos $V(X)$ para denotar o conjunto de vértices de um grafo X, e $E(X)$ para denotar o conjunto de arestas de um grafo X, não havendo necessidade de definir o par ordenado para todo grafo. No que segue, seja G um grafo simples.

Denotaremos a aresta formada pelo par $u, v \in V(G)$ por uv (ou vu), representando o par não ordenado $\{u, v\} \in E(G)$. Os vértices u e v são **extremos** da aresta uv, a aresta uv **incide** em u e em v, e os vértices u e v são **vizinhos** ou **adjacentes**.

Chamamos de **passeio** em G uma sequência de vértices $W = (v_0, v_1, \ldots, v_k)$ tal que $v_i v_{i+1} \in E(G)$, para $0 \leq i < k$. Chamamos v_0 de **início** e v_k de **término** de W, enquanto os vértices $v_1, \ldots, v_{k-1}$ são **vértices internos** de W. Um passeio W é dito **fechado** se $v_0 = v_k$.

Seja W um passeio em G. Se W não repete arestas, chamamos W de **trilha**. Se W é um passeio fechado que não repete arestas, chamamos W de **trilha fechada**. Se W não repete

vértices, chamamos W de **caminho**. Se W é um passeio fechado com pelo menos três arestas e que não repete vértices internos, chamamos W de **ciclo**.

Dados os grafos G e H, dizemos que H é **subgrafo** de G se $V(H) \subseteq V(G)$ e $E(H) \subseteq E(G)$, e denotamos essa relação por $H \subseteq G$. Se $V(H) = V(G)$, dizemos ainda que H é um **subgrafo gerador** de G.

Passeios em G também podem ser considerados como subgrafos de G. Um passeio $W = (v_0, \ldots, v_k)$ pode ser considerado como o subgrafo em que $V(W) = \{v_0, \ldots, v_k\}$ e $E(W) = \{v_0v_1, \ldots, v_{k-1}v_k\}$.

Um grafo G é dito **completo** se todo $v \in V(G)$ é adjacente a todo $u \in V(G) \setminus \{v\}$.

3 Problemas de interesse

Um primeiro problema relacionado com o TSP é o problema do Ciclo Hamiltoniano.

Problema 1 (Ciclo Hamiltoniano)**.** *Dado um grafo G, encontrar um ciclo hamiltoniano em G, isto é, um ciclo que é um subgrafo gerador em G.*

O problema do Ciclo Hamiltoniano é um clássico problema de decisão que é NP-completo [5]. Ele pode ser utilizado para provar a NP-dificuldade do TSP.

Problema 2 (TSP)**.** *Dados um grafo completo G e uma função de custo $w\colon E(G) \to \mathbb{Q}^+$, encontrar um ciclo hamiltoniano C em G que minimize $w(C) = \sum_{e \in E(C)} w(e)$.*

Infelizmente, além de ser NP-difícil, é possível mostrar que não existe algoritmo de aproximação para o TSP em geral [11]. Assim, é comum encontrar estudos que lidam com variações do TSP que impõem restrições na entrada. A mais conhecida trata de funções de custo w que respeitam a desigualdade triangular, isto é, tais que $w(uv) \leq w(ux) + w(xv)$ para todo $u, v, x \in V(G)$.

Problema 3 (TSP Métrico)**.** *Dados um grafo completo G e uma função de custo $w\colon E(G) \to \mathbb{Q}^+$ que respeita a desigualdade triangular, encontrar um ciclo hamiltoniano C em G que minimize $w(C) = \sum_{e \in E(C)} w(e)$.*

Para o TSP Métrico, existe uma 2-aproximação [10] e uma $\frac{3}{2}$-aproximação [4], sendo esta última um resultado de 1976. Desde então, não houve algoritmo com melhor fator de aproximação para o TSP Métrico. Sabe-se, ainda, que não é possível encontrar algoritmo com fator de aproximação melhor do que $\frac{123}{122}$ [7] e, portanto, também não há PTAS para o TSP Métrico.

Uma outra variante que recebeu atenção nos últimos anos é uma em que todas as arestas têm custo 1 ou 2.

Problema 4 ($(1,2)$-TSP)**.** *Dados um grafo completo G e uma função de custo $w\colon E(G) \to \{1,2\}$, encontrar um ciclo hamiltoniano C em G que minimize $w(C) = \sum_{e \in E(C)} w(e)$.*

Note que qualquer solução para o $(1,2)$-TSP terá custo entre n e $2n$, onde $n = |V(G)|$, de forma que uma 2-aproximação é trivial: basta devolver qualquer ciclo hamiltoniano. Além disso, instâncias para o $(1,2)$-TSP também respeitam a desigualdade triangular, de forma que a $\frac{3}{2}$-aproximação do TSP Métrico é diretamente válida para esse caso. Com o passar do tempo, esse fator de aproximação foi sendo reduzido: $\frac{4}{3}$, $\frac{11}{6}$, $\frac{7}{6}$ [8] e $\frac{65}{56}$ [3]. Atualmente, o melhor fator é $\frac{8}{7}$ [1]. Além disso, o melhor fator de inaproximabilidade para o $(1,2)$-TSP é de $\frac{535}{534}$ [1].

4 Objetivos

Nosso objetivo principal é complementar a formação do candidato na área de Ciência da Computação, aprofundando seu conhecimento por meio do estudo de vários algoritmos de aproximação para o $(1,2)$-TSP. Esses algoritmos utilizam diversas técnicas e envolvem diversos conceitos importantes, como os de árvore geradora mínima, emparelhamentos e 2-fator. A depender do andamento da execução do projeto, outros algoritmos de aproximação para outras variações do TSP podem ser inseridas no estudo.

5 Metodologia e viabilidade

O aluno começará estudando os dois algoritmos clássicos para o TSP Métrico, que são mais simples e didáticos, sendo ideais para a introdução a algoritmos de aproximação. Esses algoritmos encontram-se disponíveis em livros didáticos, como o livro clássico de algoritmos de aproximação de Vazirani [12], disponível na biblioteca da UFABC e também disponível online gratuitamente pelo autor. Os algoritmos para o $(1,2)$-TSP serão estudados a seguir, a começar pelos três algoritmos desenvolvidos por Papadimitriou e Yannakakis [8]. Todos os algoritmos para o $(1,2)$-TSP estão disponíveis em artigos científicos, aos quais o candidato também tem acesso pela rede da UFABC.

O candidato já possui conhecimento básico em algoritmos, programação e em teoria dos grafos, partes fundamentais para o bom desenvolvimento dessa pesquisa. Além disso, serão realizadas reuniões quinzenais entre o candidato e a orientadora para que haja exposição e discussão dos conteúdos estudados bem como estabelecimento dos caminhos a serem seguidos. O candidato deverá documentar os resultados estudados para confecção de um relatório técnico ao final do período.

Tabela 1: Cronograma das atividades planejadas.

Atividades	**Meses**											
	1	2	3	4	5	6	7	8	9	10	11	12
1	x	x										
2		x	x									
3				x	x	x						
4							x	x				
5									x	x	x	
6		x	x	x	x	x	x	x	x	x	x	x

6 Cronograma

A Tabela 1 apresenta a distribuição das atividades a seguir, previstas para realização desse projeto:

1. Compreensão dos conceitos básicos e exemplos elementares de algoritmos de aproximação;
2. Compreensão dos algoritmos para o TSP Métrico;
3. Compreensão dos três algoritmos para o $(1, 2)$-TSP de Papadimitrou e Yannakakis [8];
4. Compreensão do algoritmo de Blaser e Ram [3];
5. Compreensão do melhor algoritmo, a $\frac{8}{7}$-aproximação de Adamaszek *et al.* [1];
6. Escrita do relatório final.

Conforme o andamento do projeto, outros algoritmos para variações do TSP podem ser acrescentados aos estudos.

Referências

[1] A. Adamaszek, M. Mnich, and K. Paluch. New Approximation Algorithms for $(1, 2)$-TSP. In I. Chatzigiannakis, C. Kaklamanis, D. Marx, and D. Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi: 10.4230/LIPIcs.ICALP.2018.9.

[2] D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton Series in Applied Mathematics. Princeton University Press, 2006. ISBN 9780691129938.

[3] M. Bläser and L. S. Ram. An Improved Approximation Algorithm for TSP with Distances One and Two. In M. Liśkiewicz and R. Reischuk, editors, *Fundamentals of Computation Theory*, pages 504–515, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. doi: 10.1007/11537311_44.

[4] N. Christofides. Worst-Case Analysis of a New Heuristic for the Traveling Salesman Problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

[5] M. R. Garey and D. S. Johnson. *Computers and Intractability.* W.H. Freeman and Co., New York, 1979.

[6] R. M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, 1972. doi: 10.1007/978-1-4684-2001-2_9.

[7] M. Karpinski, M. Lampis, and R. Schmied. New inapproximability bounds for TSP. *Journal of Computer and System Sciences*, 81(8):1665–1677, 2015. doi: 10.1016/j.jcss.2015.06.003.

[8] C. H. Papadimitriou and M. Yannakakis. The Traveling Salesman Problem with Distances One and Two. *Mathematics of Operations Research*, 18(1):1–11, 1993. doi: 10.1287/moor.18.1.1.

[9] E. Pegg Jr. The icosian game, revisited. *The Mathematica Journal*, 11(3):310–314, 2009.

[10] D. J. Rosenkrantz, R. E. Stearns, and I. P. M. Lewis. An Analysis of Several Heuristics for the Traveling Salesman Problem. *SIAM Journal on Computing*, 6(3):563–581, 1977. doi: 10.1137/0206041.

[11] S. Sahni and T. Gonzalez. P-Complete Approximation Problems. *Journal of the ACM*, 23(3):555–565, 1976. doi: 10.1145/321958.321975.

[12] V. V. Vazirani. *Approximation Algorithms.* Springer-Verlag, Berlin, Heidelberg, 2001.