Rainbow Version of Dirac's Theorem: An Algorithmic Approach

Nathan Luiz Bezerra Martins **Orientadora: Yoshiko Wakabayashi**
Willian Miura Mori

Departamento de Ciência da Computação,
Instituto de Matemática e Estatística, Universidade de São Paulo

Resumo

Dada uma coleção $G = G_1, G_2, \dots, G_n$ de grafos de ordem n, definidos sobre o mesmo conjunto de vértices e que satisfazem a condição de Dirac para cada G_i, existe um G-transversal que forma um circuito hamiltoniano, também conhecido como Circuito Hamiltoniano Rainbow. Neste trabalho, desenvolvemos um algoritmo eficiente que encontra um Circuito Hamiltoniano Rainbow. Fizemos implementações tanto em `C++` quanto em `Python` e realizamos testes de desempenho para comparar as duas versões. Utilizamos a biblioteca `manim` para fazer uma animação gráfica do algoritmo.

Conceitos básicos

Definições principais:

Um grafo simples é um grafo não direcionado sem laços e sem arestas múltiplas.

Um **ciclo hamiltoniano** de G é um ciclo que visita cada vértice de G exatamente uma vez.

$\delta(G)$ é o grau mínimo de um vértice em G.

Teorema de Dirac ([?]): Se um grafo simples G com n vértices tem grau mínimo $\delta(G) \geq \frac{n}{2}$, então G contém um ciclo hamiltoniano.

Versões Rainbow de problemas clássicos

Definição:
A versão rainbow de um problema na teoria dos grafos é uma variação que adiciona a restrição de cores à solução desejada. Nesse contexto, o termo rainbow (arco-íris) refere-se a estruturas em um grafo que utilizam elementos provenientes de diferentes subconjuntos, ou arestas com diferentes rótulos ou cores, garantindo que não haja repetições.

Teorema de Dirac (*Versão Rainbow*):
Dada uma coleção $G = G_1, G_2, \dots, G_n$ de grafos de ordem n, definidos sobre o mesmo conjunto de vértices e que satisfazem a condição de Dirac para cada G_i, existe um G-transversal que forma um circuito hamiltoniano, também conhecido como Circuito Hamiltoniano Rainbow. Cada grafo G_i pode ser enxergado como se suas arestas fossem coloridas com a cor i.

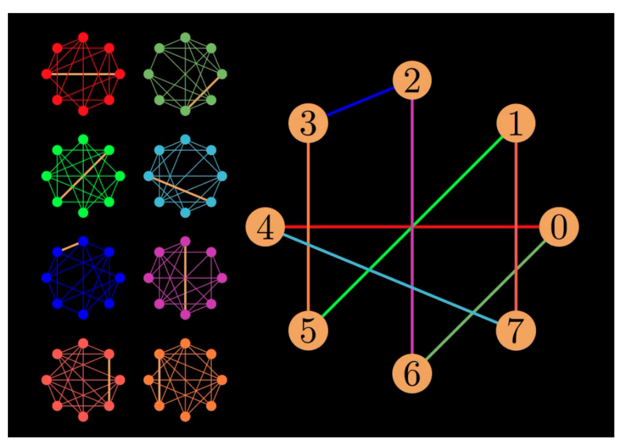


Figura 1: Exemplo de um Circuito Hamiltoniano Rainbow para uma coleção de grafos com 8 vértices.

Mais exemplos clássicos:

Floresta Geradora Mínima Rainbow: Dado um grafo G e uma coleção de cores, encontre uma floresta geradora mínima que utilize exatamente uma aresta de cada cor.

Emparelhamento Perfeito Rainbow: Dado um grafo bipartido G e uma coleção de cores, encontre um emparelhamento perfeito que utilize exatamente uma aresta de cada cor.

Fluxograma do algoritmo

A Figura **??** mostra o fluxograma do algoritmo desenvolvido. A ideia principal é, dado um objeto, que pode ser um caminho ou um ciclo, incrementar esse objeto para um objeto maior.

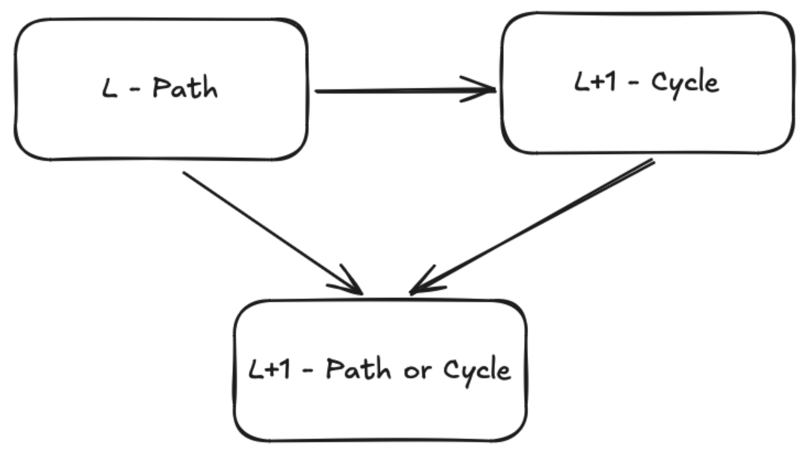


Figura 2: Fluxograma do algoritmo.

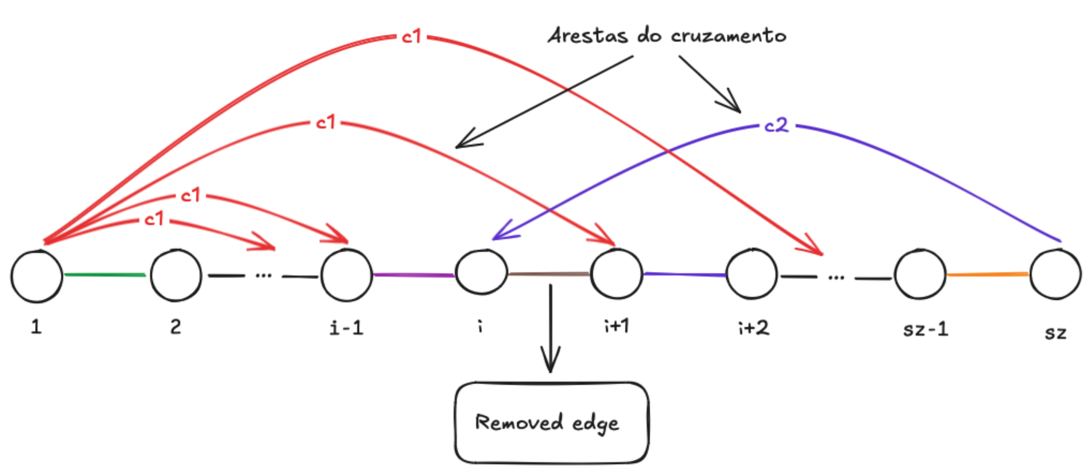
Técnicas utilizadas

Todas as técnicas utilizam fortemente a condição de Dirac para cada grafo.

Técnica do cruzamento:
Seja $P = (x_1, e_1, x_2, \dots, e_{sz}, x_{sz+1})$, sem repetição de cor e nem de vértice. Suponha que existam duas cores c_1 e c_2 que não pertencem ao caminho e não existem arestas entre x_1 e $V \setminus P$ com cor c_1 e entre x_{sz+1} e $V \setminus P$ com cor c_2. Suponha também que não existe uma aresta de cor c_1 ou c_2 entre x_1 e x_{sz+1}.
Olhando para as arestas de cor c_1 que saem de x_1, e para as arestas de cor c_2 que saem de x_{sz+1}, temos que:

$$|N_{G_{c_1}}(x_1)| + |N_{G_{c_2}}(x_{sz+1})| \geq \frac{n}{2} + \frac{n}{2} > sz - 2.$$

Pelo princípio da casa dos pombos, existe um índice i tal que as arestas (x_1, x_{i+1}) e (x_{sz+1}, x_i) são de cores c_1 e c_2, respectivamente.
Portanto, podemos formar um circuito de tamanho maior como na imagem abaixo:



Implementação

A prova feita por [?] pode ser adaptada para um algoritmo construtivo, em que o estado é um caminho ou ciclo *rainbow*, e incrementalmente aumentamos o tamanho desse caminho ou ciclo.
A complexidade do algoritmo é da ordem de $O(n^3)$, em que n é a ordem de G. Essa é complexidade é ótima, pois existem $O(n^3)$ arestas que devem ser consideradas no input.
Para validar a corretude da implementação, foram gerados grafos que respeitavam a condição de Dirac com n até 300, e o código implementado devolvia o ciclo hamiltoniano *rainbow*.

Animação

Animação demonstrando o algoritmo foram feitas com a biblioteca `manim` (do canal de Youtube `3Blue1Brown`) em Python. Um exemplo está disponível no QR Code abaixo

Informações e contato

Para mais informações, acesse a página do trabalho: `https://linux.ime.usp.br/~felipen/mac0499`
Endereço para contato:
`felipe.castro.noronha@usp.br`

Referências