Título do artigo

Hebert Rocha, Janderson Almeida, Rodrigo Oliveira

Departamento da Ciência da Computação – Universidade Federal de Roraima Pará (UFRR) – Boa Vista – RR– Brasil

jandersonatual62@gmail.com, [rodrigoarkhamcity@gmail.com](mailto:rodrigoarkhamcity@gmail.com), herbert.rocha@ufrr.com

**Abstract.** This article proposes to use the algorithm of minimum coverage of vertices, where it intends to position the minimum of traffic around the city covering as many streets as possible, resulting in cost savings for the city.

**Resumo.** Este artigo propõe a utilização o algoritmo de cobertura mínima de vértices, onde tem com a intenção de posicionar o mínimo de semáforos pela cidade abrangendo o máximo de ruas possíveis, como resultado, trazendo economia de gastos para a prefeitura.

# 1. Introdução

O artigo completo deve estar no formato apresentado neste artigo. Toda a formatação é baseada no “template” disponível no site da Sociedade Brasileira de Computação (SBC) que é utilizado nas conferencias organizadas pela SBC. O formato deve ser A4 com coluna simples, 3 cm de margem superior, 2.5 cm de margem inferior e 3.0 cm de margens laterais, sem cabeçalhos ou rodapés. A fonte principal é Times, tamanho 12, com 6 pontos de separação em cada parágrafo, exatamente como demonstrado neste parágrafo. Os números de página devem ser omitidos.

Segundo Siqueira (2011), a teoria dos grafos tem uma origem relativamente recente (século XVIII) na história da matemática, sendo seu surgimento estabelecido em 1736, ano da solução do problema das pontes de Königsberg por Euler. Além deste, poucos trabalhos surgiram até meados do século XIX, destacando-se o de Kirchhoff que, em 1847, utilizou modelos em árvores no estudo de circuitos elétricos, e o de Cayley, que utilizou o conceito de grafo para fazer a enumeração dos isômeros dos hidrocarbonetos alifáticos saturados, em química orgânica. <http://objdig.ufrj.br/60/teses/coppe_d/AngeloSantosSiqueira.pdf>

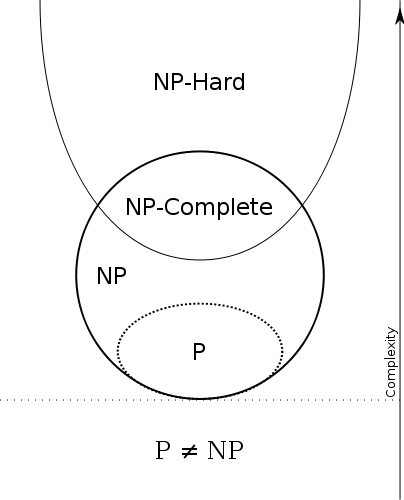
A partir daí diversos problemas surgiram dentro da teoria dos grafos, dentre elas o problema da cobertura mínima de vértices.

Uma cobertura de um grafo é qualquer conjunto de vértices que contenha pelo menos uma das pontas de cada aresta. Em outras palavras, um conjunto X de vértices é uma cobertura se toda aresta do grafo tem pelo menos uma de suas pontas em X. (IME – USP, 2013) <https://www.ime.usp.br/~pf/teoriadosgrafos/texto/TeoriaDosGrafos.pdf>

# 2. NP-completo

Informalmente, um problema está na classe NPC – e vamos nos referir a ele como um problema NP-completo – se ele está em NP e é tão “difícil” quanto qualquer problema em NP. (Cormen, 2009)

A classe NP-completo são problemas NP que possui a característica de um que um deles poder ser resolvido em tempo polinomial então todo problema NP-completo terá uma solução em tempo polinomial. (Edson Preste, 2011)



**Figura 1. Diagrama de Euler para o conjunto de problemas P, NP, NP-completo e NP-hard.**

**Fonte: Wikipédia, 2017.**

# 3. Cobertura mínima de vértice

O problema de cobertura de vértices é encontrar uma cobertura de vértices de tamanho mínimo em um dado grafo não orientado. Chamamos tal cobertura de vértices uma cobertura de vértices ótima. Esse problema é a versão de otimização de um problema de decisão NP-completo. Uma cobertura de vértices mínima é uma cobertura de vértices de menor tamanho possível. (Paulo Afonso, 2012) <https://pt.scribd.com/document/97479461/Cobertura-de-Vertices-Paulo-Afonso>

# 4. Heurística

A heurística possui uma função Cobertura Mínima de Vértices que recebe como parâmetro a variável m\_adjacencia[maxl][maxc], que corresponde à matriz de adjacência do grafo, sendo maxl o tamanho máximo de linhas e maxc o tamanho máximo de colunas.

Na linha 2, algoritmo irá executar uma repetição até que o critério de parada seja satisfeito, isso ocorre quando todas as arestas do grafo forem cobertas por um vértice. Enquanto essa condição se satisfazer, a variável do tipo int (inteiro), vmgrau armazenará o valor retornando pela função maior\_grau(), que calcula o vértice de maior grau do grafo.

Nos passos 4 à 9, a partir de um laço de repetição onde a linha assumirá um valor fixo correspondente ao vértice de maior grau (vmgrau) e a coluna variando de 1 ao tamanho máximo de colunas (maxc), o algoritmo vai zerar todas as posições da matriz relativo à vmgrau, ou seja todas as arestas que forem cobertas por este vértice receberão valor 0, o mesmo se aplica para zerar a coluna correspondente à vmgrau, no entanto a linha vai variar de 1 ao tamanho máximo de linhas (maxl) e a coluna assumirá correspondente à vmgrau. Desta forma, a matriz será composta apenas das arestas que não foram cobertas por nenhum vértice.

Após executado os laços de repetição, a variável conjunto vai receber o vértice de maior grau, ou seja o valor de vmgrau (linha 10).

Na linha 11, a função atualiza\_gvertice(), vai atualizar o valor dos vértices após cada modificação na matriz de adjacência.

|  |
| --- |
| **Algoritmo –** Cobertura mínima de vértices |

1. Dada uma matriz m\_adjacencia [maxl][maxc], sendo maxl o número máximo de linhas e maxc o número máximo de colunas.

2. **Enquanto** (critério de parada) **faça**

3. Atribui à vmgrau o valor retornado da função maior\_grau

4. **para** i indo de 1 até o número 0 na linha correspondente a vmgrau e nas colunas correspondente às variações de i.

5. A matriz recebe 0 na linha correspondente a vmgrau e nas colunas correspondente às variações de k.

6. **para** i de 1 até o número máximo de interações maxc **faça**

7. A matriz recebe valor 0 na linha correspondente a vmgrau e nas colunas correspondentes ás variações de k

8. **fim para**

9. **fim para**

10. Atribui a variável conjunto, o conjunto de U {vmgrau}

11. Atualiza o grau dos vértices.

12. **fim enquanto**

|  |
| --- |
|  |

# 5. Complexidade

A heurística possui uma função Cobertura Mínima de Vértices que recebe como parâmetro a variável m\_adjacencia[maxl][maxc], que corresponde à matriz de adjacência do grafo, sendo maxl o tamanho máximo de linhas e maxc o tamanho máximo de colunas.

# 6. Resultados

<http://www.inf.ufrgs.br/~prestes/Courses/Complexity/aula26.pdf>