**Problema da Cobertura de Vértices**  
Gabriel Ribeiro Passos

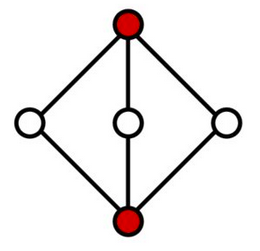
Iuri Sousa Werneck Pereira

**INTRODUÇÃO**O Problema da Cobertura de Vértices é uma questão clássica do mundo da teoria matemática, dos grafos e da ciência da computação. Em termos simples, é um problema de identificação de um conjunto de vértices (o menor possível) em um grafo onde cada aresta do grafo é incidente a pelo menos um vértice do conjunto.

Ele consiste em determinar um subconjunto S ⊂ V tal que cada arco (u, v) ∈ E incide em pelo menos um nó de S, ou seja, determinar o subconjunto de vértices no qual suas conexões abrange todas as arestas.  
  
Este problema é um conceito importante no mundo dos algoritmos de computador. Aqui estão algumas razões do porquê:

* Auxilia na compreensão das características fundamentais dos problemas NP-completos.
* Desempenhando um papel fundamental na teoria da complexidade, permite o estudo de problemas difíceis de resolver.
* Serve de base para a criação de algoritmos que tratam da resolução de problemas da vida real.

Além destes, também desempenha um papel em design de redes, bioinformática, pesquisa operacional, entre outros campos. Consequentemente, a compreensão do Problema da Cobertura de Vértices leva a uma maior compreensão da ciência da computação e ao desenvolvimento de soluções computacionais superiores.



**ENTRADA**

A entrada para este problema é qualquer grafo não ponderado tendo em vista que para se chegar ao melhor resultado, é preciso saber apenas os vértices que cobrem a maior quantidade de arestas, não importando o peso destas arestas.

**SOLUÇÃO**

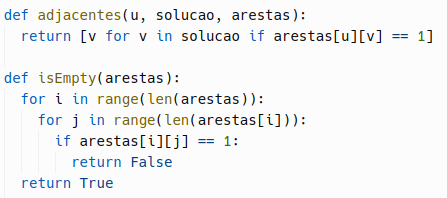
A solução deste problema será uma lista ordenada de vértices para se fazer a cobertura. Para cada vértice da solução é percorrido sua lista de adjacência eliminando as arestas de ligação entre eles. Deste modo, os últimos vértices da solução tendem a não estar na cobertura mínima dos vértices.

**RESULTADO**

O resultado esperado será um conjunto dos vértices que abrangem a maior quantidade de arestas possíveis, ou seja, encontrar uma quantidade MÍNIMA de vértices. Diante disso, o retorno será o subconjunto mínimo contendo os vértices e a quantidade de vértices que contém neste conjunto.

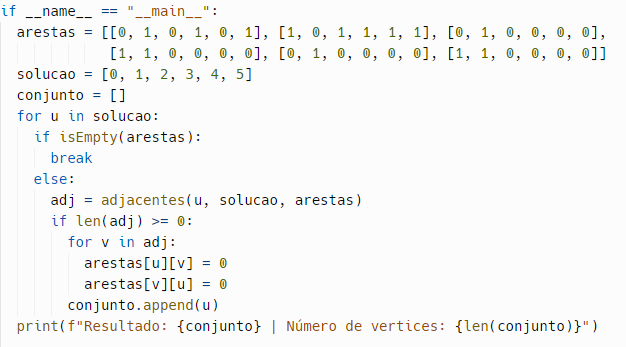
**VIZINHANÇA**

A alteração da vizinhança é feita através de trocas(swap). A quantidade de trocas feitas na solução, é baseado em uma porcentagem pré-definida. Tomando uma porcentagem definida de 10% e um número de 1100 vértices: neste caso, serão feitas onze trocas. Por fim, será retornada uma nova solução.

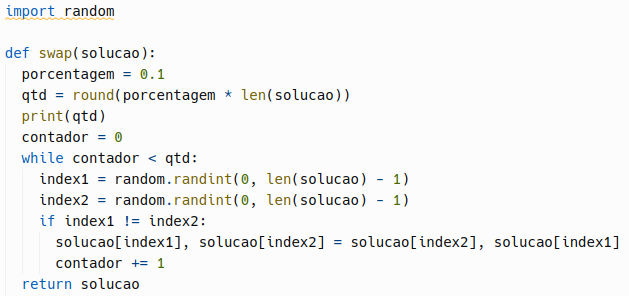


A função ‘adjacentes’ retorna uma lista de vértices adjacentes ao vértice (u).

A função ‘isEmpty’ retorna False caso exista alguma aresta a ser coberta e True caso todas as arestas já tenham sido cobertas.



Este trecho do código percorre a lista de solução verificando se existe alguma aresta para ser coberta. Caso exista, faz a cobertura dos vértices adjacentes removendo a aresta de ida e a de volta. Não havendo mais nenhum vértice adjacente a (u), ele é adicionado na lista (conjunto). Caso não exista mais nenhuma aresta a ser coberta, o laço de repetição é finalizado. Ao final é impresso o conjunto de vértices que realizam a cobertura do grafo (conjunto) e o número de vértices contidos nesse conjunto.



A função ‘swap’ realiza a troca da ordem dos vértices contidos na solução. Esta troca é baseada em uma porcentagem do tamanho da lista de solução. No caso do exemplo acima, esta porcentagem foi definida em dez por cento. Após isso, é escolhido randomicamente dois índices para a realização do swap. Caso o número de trocas ainda não tenha sido atingido, o processo se repete. Ao final, a função retorna a nova lista de solução.

https://snap.stanford.edu/data/