

title

Hugo Camposs sss Veríssimo

Abstract – abstrato em pt bla bla ns q in English**Resumo** – resumo baksdakdjsa in Portuguese**Keywords** – keywords Note: in English (optional)**Palavras chave** – keywords pt in Portuguese (optional)

I. INTRODUÇÃO

Atualmente, os problemas em grafos são amplamente estudados, pelo facto de terem a capacidade de modelar diversas situações reais, desde as mais palpáveis, como redes de computadores (problema *Minimum Spanning Tree*) até às mais abstratas, como física teórica (problema *Maximum Weight Cut*) [1].

Este relatório visa explorar o problema *Maximum Weight Cut*, conhecido em português por Corte de Peso Máximo, que consiste na divisão de um grafo não direcionado, $G(V, E)$, onde $|V| = n$ vértices e $|E| = m$ arestas de peso $w_{i,j} \geq 0 \forall (i, j) \in E$, em dois subconjuntos complementares, S e T , de forma a maximizar a soma dos pesos das arestas que ligam os dois conjuntos [2], isto é

$$\max \sum_{i \in S, j \in T} w_{i,j} \\ \begin{cases} S \cup T = V \\ S \cap T = \emptyset \end{cases}$$

Apesar do problema oposto, conhecido como *Minimum Weight Cut*, ter um algoritmo de resolução em tempo polinomial, em certas condições, o problema *Maximum Weight Cut* não o possui, sendo um problema *NP-Hard*. Isto implica que à medida que o tamanho do grafo aumenta, encontrar soluções exatas para este problema tornam-se computacionalmente caras [1].

Ao longo deste relatório, serão abordadas duas estratégias para a resolução do problema: uma busca exaustiva e uma heurística gulosa.

II. ORG E CODIGO

ns q gerar graficos, ficheiro no github?
etc etc

III. ALGORITMO DE PESQUISA EXAUSTIVA

Tendo em conta o algoritmo de pesquisa exaustiva, este visa gerar todas as combinações possíveis de subconjuntos de V e, para cada subconjunto, calcular o peso do corte e comparar com o melhor corte encontrado até ao momento, ou seja, o algoritmo tem duas

fases: a geração de todos os subconjuntos possíveis e a avaliação de cada subconjunto.

Este algoritmo garante a obtenção da solução ótima, no entanto, o seu custo computacional é exponencial, sendo impraticável para grafos de grande dimensão.

... Este algoritmo pode ser traduzido em pseudocódigo da seguinte forma:

Algorithm 1 Exhaustive Search (graph)**Input:** matrix adjacencia do grafo**Output:** s, t, weight cut value

```

1: input_set ← {0, 1, ..., len(graph) - 1}
2: subsets ← EMPTY LIST
3: n ← LENGTH OF input_set
   ▷ Generate all subsets
4: for r from 0 to n do
5:   for each subset in combinations(input_set, r)
6:     do
7:       Add subset to subsets
8:   end for
9: end for
10: best ← input_set
11: weight ← 0
   ▷ Evaluate each subset
12: for each S in subsets do
13:   new_weight ←
14:     CALCULATE_CUT_WEIGHT(graph, S)
15:   if new_weight > weight then
16:     best ← S
17:     weight ← new_weight
18:   end if
19: end for
20: S ← best
21: T ← input_set - best
22: return S, T, weight

```

IV. ALGORITMO DE PESQUISA GULOSA

técnica heurística gulosa (greedy), que procura uma solução aproximada de maneira eficiente, tomando decisões locais que parecem melhores no momento, embora sem garantia de que serão as ótimas globais.

V. QQCENA

jdsauhdisa aa a a

VI. QQCENA

dasdsad sn q disse

REFERENCES

- [1] Wikipedia contributors, “Maximum cut — Wikipedia, the free encyclopedia”, https://en.wikipedia.org/w/index.php?title=Maximum_cut&oldid=1247744814, 2024, [Online; accessed 27-October-2024].
- [2] Noga Alon, Béla Bollobás, Michael Krivelevich, and Benny Sudakov, “Maximum cuts and judicious partitions in graphs without short cycles”, *Journal of Combinatorial Theory, Series B*, vol. 88, no. 2, pp. 329–346, 2003.