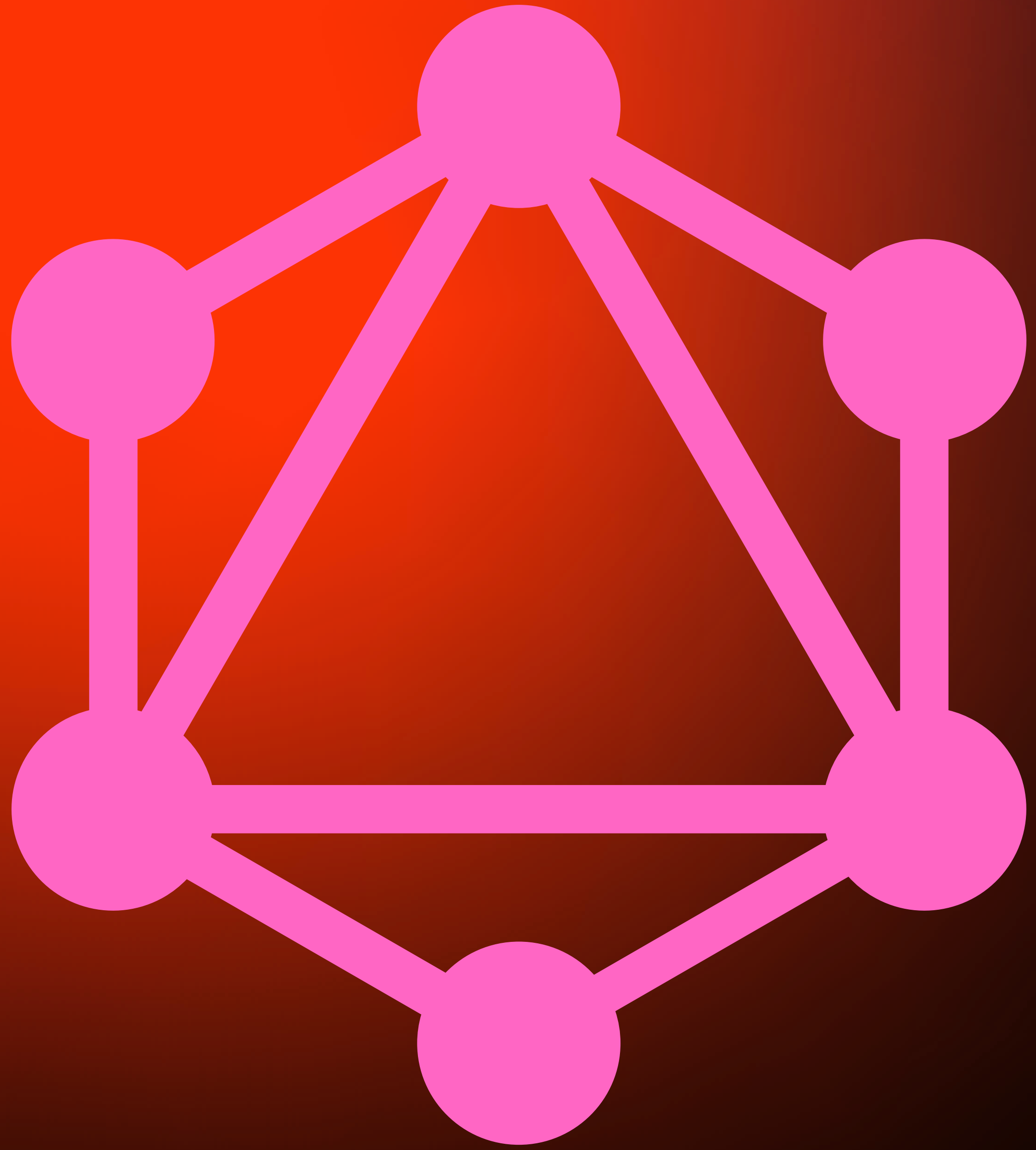
Approximate Graph Coloring

Alunos: Ramsés Carvalho e Jorge Serrão



Artigo Karger, Motwani, Sudan, 1998.

Approximate Graph Coloring by Semidefinite Programming

No artigo é estudado o problema de colorir k – grafos com o menor número possível de cores, para solucionar este problema é posto um algoritmo de tempo polinomial aleatório, que colore um grafo de 3 cores em n vértices com um mínimo de cores {O(D^(1/3) log^(1/2) Dlog n),O(n^(1/4) log^(1/2) n)} onde D é o grau máximo de qualquer vértice.

Semicoloração: Um k-semicolorizável de um grafo G é uma atribuição de k cores para pelo menos metade de seus vértices de forma que não haja dois vértices adjacentes com a mesma cor. Se um algoritmo A pode ki-semicolorir qualquer subgrafo i-vértice do grafo G em tempo randomizado polinomial, onde ki aumenta com i, então A pode ser usado para O(kn log n)-cor G.

Artigo Karger, Motwani, Sudan, 1998.

Approximate Graph Coloring by Semidefinite Programming

Arredondamento por partições de hiperplano: Considera-se um hiperplano H. É dito no artigo que para separar dois vetores se eles não estão no mesmo lado do hiperplano. Para qualquer aresta {i,j} ∈ E, nós dizemos que o hiperplano H corta a beira se ele separa o vetor vi e vj. Usando o algoritmo de Wigderson o algoritmo pode ser melhorado passando de O(n^0,613) cores para O(n^0,387) cores.

Teoria da dualidade: por definição dado um grafo G = (V, E) em n vértices, um vetor estrito de coloração k de G é uma atribuição de vetores unitários ui do espaço R^n para cada vértice i ∈ V, de modo que para quaisquer dois vértices adjacentes i e j o produto escalar de seus vetores satisfaz a igualdade (ui, uj) = -1/(k1).

Artigo Karger, Motwani, Sudan, 1998.

Approximate Graph Coloring by Semidefinite Programming

O segundo teorema do capitulo 9 estabelece que os grafos têm uma grande lacuna entre seu vetor de número cromático e os números cromáticos reais.

Seja n=(n | r) denotam o número de vértices do grafoK(m,r,t). Para r = m/2 e t=m/8, o grafo K(m,r,t) é vetor de 3 cores, mas tem um número cromático de pelo menos n^0,0113.

X >= (1,007864)^lg n = n^lg1,007864 ≈ n^0,0113

Artigo Karger, Motwani, Sudan, 1998.

Approximate Graph Coloring by Semidefinite Programming

O terceiro teorema fala que existe um grafo kneser $K(m,r,t)$ que é um vetor de 3 cores mas tem um número cromático excedendo $n^{0,016101}$, onde $n = (m \mid n)$ denota o número de vértices no grafo.

Usando o teorema de Milner é possível provar que o expoente do número cromático é pelo menos.

$$1- \frac{(m - t)\log 2m/(m - t) + (m + t)\log 2m/(m +t)}{2((m - r)\log m/(m - r) + r \log m/r)}$$

Algoritmo Guloso

(Somatorio(Σ) de cr=1 até v)*((Somatorio(Σ) de i=adj[u].begin() até adj[u].end()) + (Somatorio(Σ)de cr=1 até v) + (Somatorio(Σ) de i=adj[u].begin() até adj[u].end()))

(Somatorio(Σ) de cr=1 até v)*(adj[u].end() - adj[u].begin() +1) + (Somatorio(Σ) de cr=1 até v)*v + (Somatorio(Σ) de cr=1 até v)*(adj[u].end() - adj[u].begin() +1)

vadj[u].end() - vadj[u].begin() + v + v^2 + vadj[u].end() - vadj[u].begin() + v

v^2 + 2vadj[u].end() - 2vadj[u].begin() + 2v

O(v^2) ou O(n^2)

Algoritmo Backtracking

(Somatorio(Σ) de i=1 até v) * (somatorio(Σ) de j=i+1 até v) + T(i) = { 1 i=v | T(i+1) +1 i < v }

Parte 1

(somatorio(Σ) de i=1 até v) * (v-i)
v(v+1)/2 -> (v^2)/2 + v/2
(v^2) – (v^2)/2 + v/2
(v^2)/2 + v/2

Artigo Karger, Motwani, Sudan, 1998.

Approximate Graph Coloring by Semidefinite Programming

$$T(i) = T(i+1) + 1$$

$$T(i) = [T(i+2) + 1] + 1$$

$$T(i) = T(i+2) + 2$$

$$T(i) = [T(i+3) + 1] + 2$$

$$T(i) = T(i+3) + 3$$

$$T(i) = T(i+k) + k$$

Assume: $i+k = v$ logo $k = v - i$

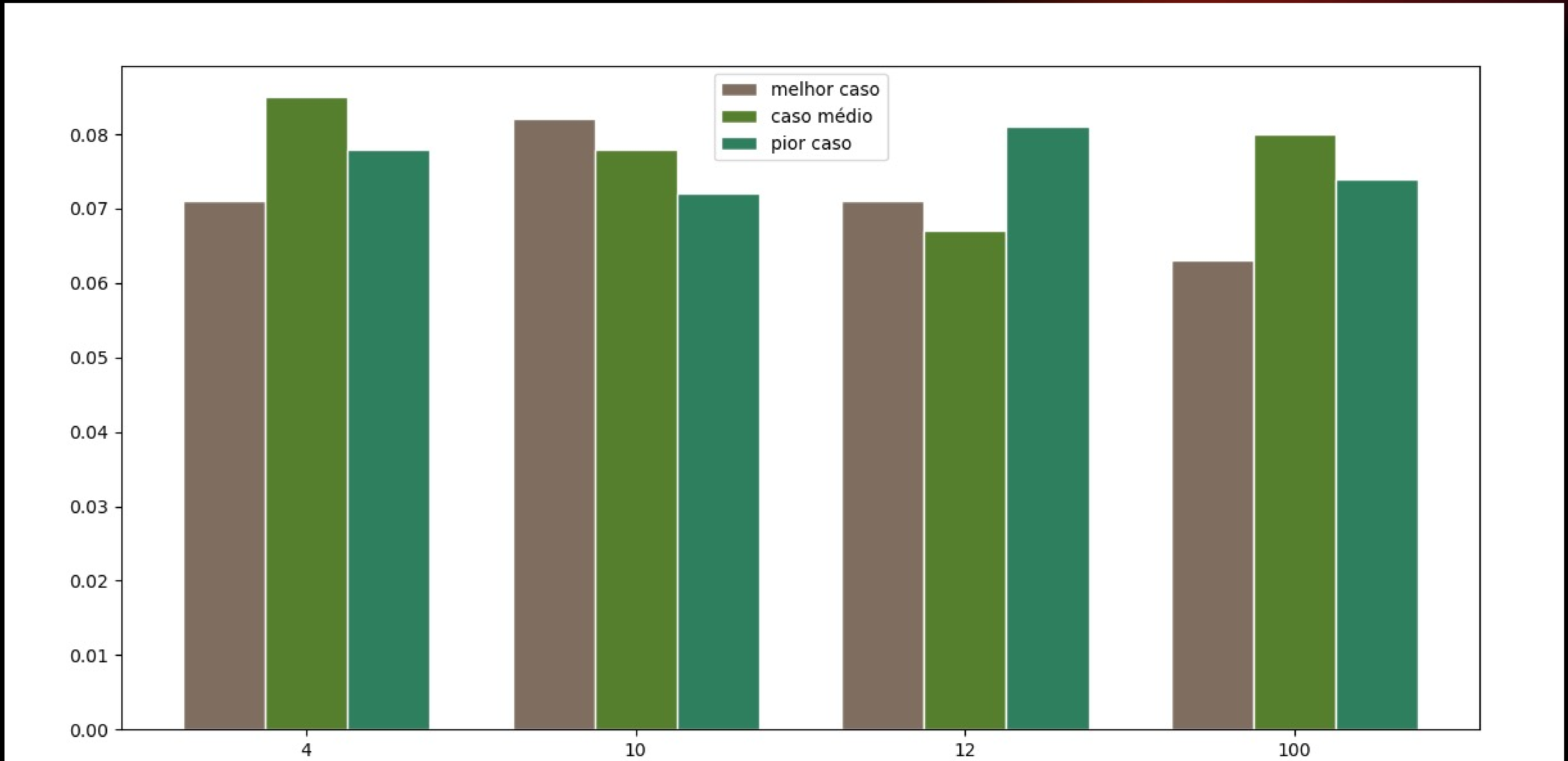
Casos de Teste

- Melhor caso: quando nenhum vértice é conectado a alguma aresta, assim só haverá uma cor.
- Pior caso: quando cada vértice do grafo é conectado com cada um dos outros vértice, assim haverá uma cor diferente para cada vértice.
- Caso médio: foi gerado um grafo que conecta os vértices aleatoriamente, e com a metade da quantidade de arestas de cada do pior caso.

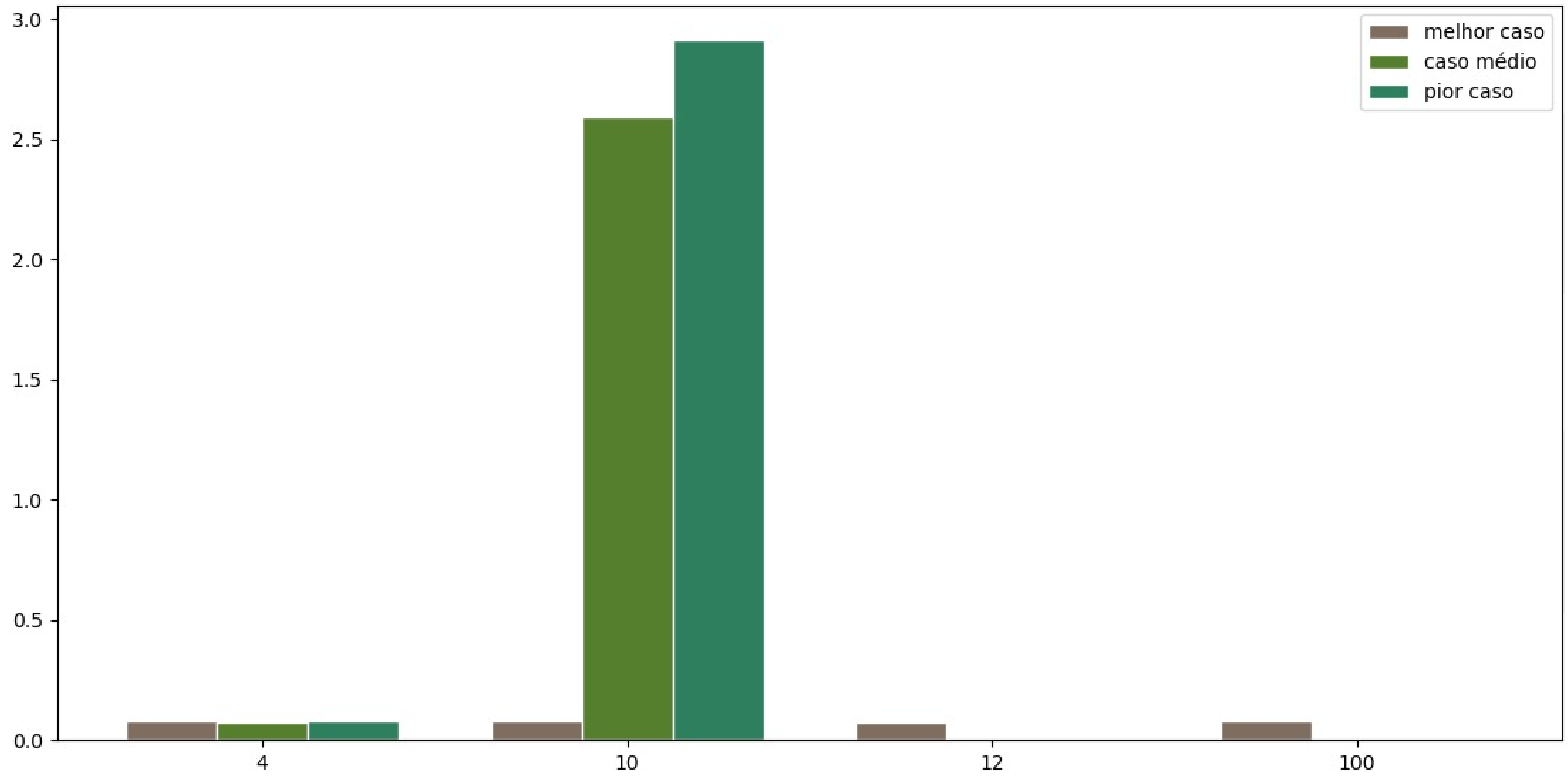
Resultados

Test e N°	Vértices	Arestas (pior/ médio)	Algoritmo Guloso: Tempo - N° Cores (melhor/ pior/ médio)	Algoritmo BackTracking: Tempo - N° Cores (melhor/ pior/ médio)
1	4	16/8	0.071 s - 1 cor/ 0.078 s - 4 cores/ 0.085 s - 3 cores	0.079 s - 1 cor/ 0.076 s - 4 cores/ 0.075 s - 2 cores
2	10	100/50	0.082 s-1 cor/0.072 s - 10 cores/ 0.078 s - 5 cores	0.078 s - 1 cor/2.913 s - 10 cores/ 2.592 s - 5 cores
3	12	144/72	0.071 s-1 cor/0.081 s - 12 cores/ 0.067 s - 5 cores	0.071 s - 1 cor/ timelimitout/ timelimitout
4	100	10^4/5x10^3	0.063 s-1cor/0.074s - 100 cores/ 0.080 s - 27 cores	0.077 s - 1 cor/ timelimitout/ timelimitout
5	10000	10^8/10^7	0.081 s-1 cor/ alloc error/ 18.677s - 294 cores	execution error/ execution error/ execution error

Algoritmo Guloso



Algoritmo backtracking



Obrigado