LIII Simpósio Brasileiro de Pesquisa Operacional
João Pessoa - PB, 3 a 5 de novembro de 2021

HEURÍSTICA PROBABILÍSTICA GUIADA PELOS POTENCIAIS DOS VÉRTICES PARA O PROBLEMA DO CORTE MÁXIMO

Pablo Luiz Braga Soares
Universidade Federal do Ceará – Campus de Russas
Rua Felipe Santiago – Nº 411, Cidade Universitária, Russas/CE, CEP 62900-000
pablo.soares@ufc.br

José Arimateia Fabricio de Castro Filho
Universidade Federal do Ceará – Campus de Russas
Rua Felipe Santiago – Nº 411, Cidade Universitária, Russas/CE CEP 62900-000
filho.arimateia@hotmail.com

RESUMO

O Problema do Corte Máximo (*Max-Cut*) consiste em dividir os vértices de um grafo em dois conjuntos de forma que o somatório dos pesos das arestas entre esses dois conjuntos seja o maior possível. O problema é NP-Difícil e no melhor do nosso conhecimento ainda não existe um método ou trabalho na literatura que tenha conseguido obter o valor ótimo de instâncias com tamanhos grandes. Neste trabalho, propomos uma nova heurística que é dada pela combinação de uma busca local, guiada pelos potenciais internos e externos dos vértices, com o uso de probabilidade para criar uma solução inicial. Apresentamos experimentos computacionais em 54 instâncias conhecidas da literatura e outras instâncias criadas. Os resultados obtidos são comparados com um trabalho da literatura que é considerado o melhor dentre aqueles que também usam probabilidade para compor um método de resolução para o *max-cut*.

PALAVRAS CHAVE. Problema do Corte Máximo, Heurística probabilística, Potencial do vértice.

Metaheurísticas (MH), Teoria e Algoritmos em Grafos (TAG), Estatística e Modelos Probabilísticos (EST&MP)

ABSTRACT

The Max Cut Problem consists of dividing the vertices of a graph into two sets so that the sum of the edge weights between these two sets is as high as possible. The problem is NP-Hard and to the best of our knowledge there is still no method or work in the literature that has managed to obtain the optimal value of instances with large sizes. In this work we propose a new heuristic that is given by combining a local search, guided by the internal and external potentials of the vertices, with the use of probability to create an initial solution. We present computational experiments on 54 instances known from the literature and other created instances. The results obtained are compared with a work in the literature that is considered the best among those that also use probability to compose a resolution method for the max cut.

KEYWORDS. Max Cut Problem, Probabilistic heuristic, Vertex potential.

Metaheuristics (MH), Graphs algorithms and theory (GAT), Statistics and probabilistic models (ST&PM)

1. Introdução

De acordo com Soares [2018], o problema do corte máximo (*Max-Cut*) pode ser descrito utilizando um grafo não direcionado $G = (V, E)$, com $|V| = n$ vértices e $|E| = m$ arestas ponderadas. Seja $c_{ij} = c_{ji}$ o peso de cada aresta $\{i, j\} \in E$, $c_{ij} = 0$ para $\{i, j\} \notin E$, e $c_{ii} = 0$, para todo $1 \leq i \leq n$. Qualquer partição $(S, \overline{S} := V \backslash S)$ dos vértices de V define um corte de G, qual seja, o subconjunto de arestas com uma extremidade em S e a outra em $\overline{S}$, onde S ou $\overline{S}$ pode ser vazio. O problema do *Max-Cut* consiste então em encontrar $(S, \overline{S})$ tal que a soma dos pesos das arestas do corte seja máximo. É importante destacar que, tomando $c_{ij} = 0$ para $\{i, j\} \notin E$, pode-se sempre admitir que o grafo G é completo.

O desenvolvimento de algoritmos voltados para o *Max-Cut* tem sua necessidade, pois são muitas as aplicações práticas e problemas matemáticos que são modelados como *Max-Cut*, tais como o física estatística, segmentação de imagens, fluxos em redes e entre outras demais aplicações [Barahona et al., 1988; Boykov e Jolly, 2001; Liers et al., 2011].

Sendo classificado como um problema NP-Difícil [Karp, 1972], torna-se um desafio computacional obter o valor do corte máximo até mesmo para instâncias de tamanho intermediárias, grafos com aproximadamente 150 vértices [Krislock et al., 2014]. Portanto, para instâncias maiores, é necessário utilizar abordagens que forneçam soluções com boa qualidade em tempo aceitável.

Dentre as abordagens mais recentes aplicadas ao *Max-Cut*, destacam-se as meta-heurísticas baseadas em população, como algoritmos genéticos [Kim et al., 2019], baseadas em memória adaptativa em *tabu search* [Kochenberger et al., 2013], baseadas em *simulated annealing* [Myklebust, 2015], baseadas em busca local [Benlic e Hao, 2013], baseados em algoritmos probabilísticos [Wang et al., 2013] e versões híbridas que mesclam estas e outras abordagens [Arráiz e Olivo, 2009; Bouhmala, 2019].

Esse trabalho apresenta uma heurística para o *Max-Cut* que combina probabilidade, busca local e atribuição dos vértices aos conjuntos S ou $\overline{S}$ com base no potencial interno e externo que um vértice possui. A ideia do potencial foi elaborada com base em condições de otimalidade para fixação de variáveis em um *branch-and-bound* desenvolvido em Soares [2018].

Para avaliar o desempenho e a competitividade da heurística em termos de qualidade e eficiência da solução, realizamos experimentos computacionais em 54 instâncias da literatura e comparamos os resultados com o trabalho de Wang et al. [2013] que é o melhor trabalho para o *Max-Cut* que também é baseado em probabilidade. O restante desse trabalho está organizado da seguinte forma. A seção 2 descreve a ideia principal dos potencias dos vértices. A seção 3 mostra como a ideia dos potenciais foi usada através do pseudocódigo da heurística. A seção 4 é dedicada aos experimentos e resultados computacionais. E finalmente na seção 5 as considerações e direções futuras serão apresentadas.

2. Definições Preliminares

Na seção 2.1 será definido os potenciais totais, internos e externos de um vértice. Na seção 3 será mostrado o algoritmo em forma de pseudocódigo da heurística informando como a ideia do potencial foi utilizada como critério de guiar a busca local e onde aplicamos o não determinismos da heurística.

2.1. Potencias do vértice

Seja $G = (V, E)$, um grafo não direcionado com $|V| = n$ vértices e $|E| = m$ arestas ponderadas, tal que $c_{ij} = c_{ji}$ é o peso de cada aresta $\{i, j\} \in E$, para todo $1 \leq i < j \leq n$.

Definição 1 *Seja $k \in V$, o potencial total do vértice k é dado por*

$$C_k = \sum_{\substack{j \in V \\ j \neq k}} c_{kj}. \tag{1}$$

Definição 2 *Seja $(S, \overline{S})$ uma partição dos vértices de V e sejam $k \in S$ e $p \in \overline{S}$. Os potencias interno dos vértices k e p são dados por*

$$C_k^{int} = \sum_{\substack{j \in S \\ j \neq k}} c_{kj}, \quad C_p^{int} = \sum_{\substack{j \in \overline{S} \\ j \neq p}} c_{pj}. \tag{2}$$

Definição 3 *Seja $(S, \overline{S})$ uma partição dos vértices de V e sejam $k \in S$ e $p \in \overline{S}$. Os potencias externo dos vértices k e p são dados por*

$$C_k^{ext} = \sum_{\substack{j \in \overline{S} \\ j \neq k}} c_{kj}, \quad C_p^{ext} = \sum_{\substack{j \in S \\ j \neq p}} c_{pj}. \tag{3}$$

A Figura 1 $a)$ apresenta um grafo $G = (V, E)$ não direcionado com 6 vértices e 8 arestas ponderadas. A Figura 1 $b)$ mostra um exemplo de uma possível partição dos vértices de V em $S = \{1, 3, 5\}$ e $\overline{S} = \{2, 4, 6\}$, onde as arestas pontilhadas representam as arestas que fazem parte do corte gerado por $(S, \overline{S})$. Os potenciais do vértice 1 são dados por $C_1 = c_{12} + c_{13} + c_{15} = 2 + 1 + 3 = 6$, $C_1^{int} = c_{13} + c_{15} = 1 + 3 = 4$ e $C_1^{ext} = c_{12} = 2$. A Tabela 1 sumariza os potenciais totais, internos e externos dos vértices levando em consideração a partição $(S, \overline{S})$ da Figura 1 $b)$.

Tabela 1: Potencias totais, internos e externos levando em consideração a partição $(S, \overline{S})$ da Figura 1 $b)$

Vértice	C	C^{int}	C^{ext}
1	6	4	2
2	4	1	3
3	4	3	1
4	5	0	5
5	10	5	5
6	1	1	0

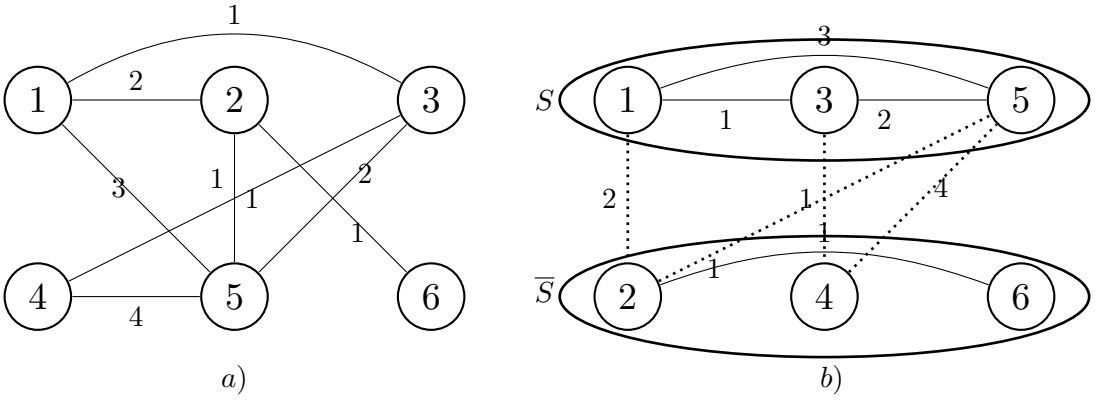


Figura 1: $a)$ Grafo não direcionado com $|V| = 6$, $|E| = 8$ e $b)$ Partição do vértices de V em $S = \{1, 3, 5\}$ e $\overline{S} = \{2, 4, 6\}$

3. Heurística Probabilística

O algoritmo mostrado a seguir descreve a forma de aplicação das principais ideias do uso dos potenciais dos vértices na busca local e do não determinismo. O algoritmo recebe como entrada o conjunto de vértices do grafo e como saída temos uma partição dos vértices de V em $(S, \overline{S})$. Inicialmente os conjuntos S e $\overline{S}$ iniciam vazios, linha 3. Na linha 4 um vértice de V é escolhido de forma aleatória e na linha 5 o vértice escolhido é atribuído ao conjunto $\overline{S}$, sendo os outros $n-1$ vértices de V atribuídos ao conjunto S. Na linha 6 são calculados os valores de C_k^{int} e C_k^{ext} $\forall k \in V$ de acordo com as definições 2 e 3. Entre as linhas 8 e 14 iremos selecionar o maior valor de Δ_k, ou seja, a maior diferença entre C_k^{int} e C_k^{ext} $\forall k \in V$. Da mesma forma será armazenado em i o vértice k que resultou no maior valor de Δ. Note que, um valor de $\Delta_k > 0$ significa que o vértice k irá contribuir com o aumento do valor do corte ao ser trocado do conjunto S para $\overline{S}$ ou vice-versa. Portanto caso exista algum vértice com valor $\Delta > 0$, o vértice em questão será permutado entre os conjuntos, o que irá gerar um aumento no valor do corte e uma nova partição $(S, \overline{S})$, essas operações são realizadas entre as linhas linhas 15 e 22. Finalmente na linha 23 será recalculado os valores de C_k^{int} e C_k^{ext} $\forall k \in V$ após a permuta do vértice que tinha o maior valor de Δ.

Algorithm 1 Heurística probabilística guida pelos potenciais

1: **Entrada**: V
2: **Saída**: $(S, \overline{S})$
3: $S \leftarrow \emptyset \quad \overline{S} \leftarrow \emptyset$
4: $a \leftarrow$ vértice de V escolhido aleatoriamente
5: $\overline{S} \leftarrow \overline{S} \cup \{a\} \quad S \leftarrow V \backslash \{a\}$
6: Calcular C_k^{int} e C_k^{ext} $\forall k \in V$
7: **repeat**
8: $\quad \max \leftarrow 0 \quad i \leftarrow a$
9: $\quad$ **for** $k \leftarrow 1, \ldots, |V|$ **do**
10: $\quad\quad \Delta_k \leftarrow C_k^{int} - C_k^{ext}$
11: $\quad\quad$ **if** $\Delta_k > \max$ **then**
12: $\quad\quad\quad \max \leftarrow \Delta_k \quad i \leftarrow k$
13: $\quad\quad$ **end if**
14: $\quad$ **end for**
15: $\quad$ **if** $\max > 0$ **then**
16: $\quad\quad$ **if** $i \in \overline{S}$ **then**
17: $\quad\quad\quad S \leftarrow S \cup \{i\}$
18: $\quad\quad\quad \overline{S} \leftarrow \overline{S} \backslash \{i\}$
19: $\quad\quad$ **else**
20: $\quad\quad\quad \overline{S} \leftarrow \overline{S} \cup \{i\}$
21: $\quad\quad\quad S \leftarrow S \backslash \{i\}$
22: $\quad\quad$ **end if**
23: $\quad\quad$ Atualizar C_k^{int} e C_k^{ext} $\forall k \in V$
24: $\quad$ **end if**
25: **until** $\max > 0$

Nota-se que a cada nova execução da heurística, uma nova partição inicial de $(S, \overline{S})$ pode ser gerada, com probabilidade de $\frac{1}{n}$, de acordo com o vértice sorteado no início para ser atribuído a $\overline{S}$. Vale notar que a quantidade máxima de partições $(S, \overline{S})$ diferentes é dada por $2^{(n-1)}$, onde n é a

quantidade de vértices. Esse fato ocorre devido a termos partições espelhadas que geram o mesmo valor de corte. Ao observar a Figura 1 b) nota-se que ao trocarmos todos os vértices de S pelos vértices de $\overline{S}$ e vice-versa ainda teríamos o mesmo valor de corte.

4. Experimentos e resultados computacionais

Na subseção 4.1 são descritas dois conjuntos de instâncias utilizadas nos experimentos computacionais. Já na subseção 4.2 será mostrado os parâmetros utilizados na execução da heurística, assim como a configuração do ambiente utilizado. As subseções 4.3 e 4.4 são dedicadas aos experimentos realizados e resultados obtidos para cada conjunto de instância.

4.1. Instâncias

O primeiro conjunto de instâncias utilizado foi criado pelos autores com o intuito de realizar teste preliminares e assim ajustar os critérios de parada da heurística com base nos resultados obtidos. Foram criadas 18 instâncias distribuídas nos seguintes tamanhos de vértices 15, 20, 25 e 28 e dividias em três grupos com quantidades iguais K, R e B. As instâncias do tipo K, são grafos completos e com pesos inteiros nas arestas variando entre 1 e 10. As instâncias R e B foram criadas para possuírem pelo menos 60% de densidade com pesos inteiros nas arestas variando entre 1 e 10 para instâncias R e valor 0 ou 1 para instâncias B.

O segundo conjunto de instâncias contém 54 instâncias denominadas $G_1, \ldots, G_{54}$ com tamanho variando entre $800 - 3000$ vértices e está disponível em `https://grafo.etsii.urjc.es/optsicom/maxcut/`. Esse conjunto de instâncias é composto de grafos toroidais, planares e aleatórios com valores de peso variando entre -1, 0 ou 1 e foram criadas por Helmberg e Rendl [2000] usando um gerador de grafos independente denominado *rudy*.

4.2. Configuração do ambiente e parâmetros de execução

A heurística foi implementada no ambiente de desenvolvimento DEV-C++ utilizando a linguagem de programação C, em uma máquina com processador Intel Core i3-9100F com 8GB de RAM e sistema operacional Windows 10.

4.3. Experimento 1

Neste experimento utilizamos as instâncias K, R e B. A heurística, foi executada 5 vezes para cada instância com os seguintes critérios de parada: duração máxima de tempo execução em 1800 segundos; um limite de execuções estabelecido pela quantidade de vértices do grafo adicionado de 500 unidades; ou até que não fosse obtido nenhum valor $\Delta_k > 0$, conforme algoritmo descrito na seção 3.

A Tabela 2 é composta de 7 colunas. A coluna 1 apresenta o nome da instância, a coluna 2 informa a quantidade de vértices da instância, a coluna 3 mostra o valor obtido pela heurística H_v, a coluna 4 o valor do corte ótimo, referenciado aqui por O_v, na coluna 5 mostramos o $GAP = \frac{O_v - H_v}{O_v} \times 100$, em porcentagem, entre o valor ótimo e o valor retornado pela heurística, e nas colunas 6 e 7 o tempo de execução da heurística e do *branch-and-bound* respectivamente. O valor ótimo dessas instâncias foi obtido através da implementação de um *branch-and-bound* capaz de encontrar a solução ótima em grafos com até 30 vértices no limite de 1800 segundos.

Os resultados da Tabela 2 mostram que a heurística foi capaz de encontrar a solução ótima de todas as instâncias testadas em menos de 0.5s de execução. Vale destacar que para as instâncias de tamanho 15, o *branch-and-bound* encontrou a solução ótima em menos tempo do que a heurística. No entanto, nota-se que a medida que são executadas instâncias com tamanhos maiores, o tempo de execução do *branch-and-bound* tende a aumentar obtendo um tempo médio de 44.48 segundos, enquanto que a heurística tende a manter o mesmo tempo de execução para esse conjunto de instâncias com média de 0.35 segundos.

Tabela 2: Comparação dos resultados obtidos pela heurística nas instâncias K, R e B

Instância	$\|V\|$	H_v	O_v	$GAP(\%)$	Tempo H_v	Tempo *Branch-and-Bound*
$K.1$	15	292	292	0.00	0.35	0.02
$K.2$	15	334	334	0.00	0.36	0.02
$K.1$	25	868	868	0.00	0.35	12.65
$K.2$	25	925	925	0.00	0.37	12.55
$K.1$	28	1113	1113	0.00	0.35	120.84
$K.2$	28	1153	1153	0.00	0.39	120.67
$R.1$	20	571	571	0.00	0.34	0.28
$R.2$	20	548	548	0.00	0.35	0.29
$R.1$	25	809	809	0.00	0.36	12.60
$R.2$	25	830	830	0.00	0.36	12.55
$R.1$	28	969	969	0.00	0.35	120.68
$R.2$	28	1046	1046	0.00	0.34	120.74
$B.1$	15	35	35	0.00	0.35	0.03
$B.2$	15	38	38	0.00	0.35	0.02
$B.1$	25	89	89	0.00	0.36	12.57
$B.2$	25	100	100	0.00	0.37	12.56
$B.1$	28	123	123	0.00	0.35	120.67
$B.2$	28	113	113	0.00	0.38	120.81
média					0.35	44.48

4.4. Experimento 2

Neste experimento utilizamos o conjunto de instâncias da literatura denominadas de instâncias G. A heurística, foi executada 5 vezes para cada instância com os seguintes critérios de parada: duração máxima de tempo execução em 1800 segundos (TL); um limite de execuções estabelecido pela quantidade de vértices do grafo adicionado de 500 unidades (LE); ou até que não fosse obtido nenhum valor $\Delta_k > 0$, conforme algoritmo descrito na seção 3.

A Tabela 3 apresenta nas colunas 1 e 2 o nome e o tamanho da instâncias, a coluna 3 mostra o melhor valor obtido pela heurística dentre as 5 execuções, as colunas 4 e 5 mostram o desvio padrão (σ) e a média obtida por instância, e as colunas 6 e 7 identificam o tempo médio nas 5 execuções e qual o critério de parada que mais foi ativado nas execuções respectivamente. Os três melhores valores de σ foram obtidos nas instâncias G_{32}, G_{48} e G_{49} e o pior valor 30.07 foi obtido pela instância G_{26}.

No melhor do nosso conhecimento, ainda não existe um método ou trabalho na literatura que tenha conseguido obter o valor ótimo das instâncias G. Dessa forma os resultados obtidos pela heurística desenvolvida nesse trabalho foram comparados com o trabalho de Wang et al. [2013] que é um algoritmo GRASP - *Greedy randomized adaptive search procedure* combinado com um procedimento de busca tabu.

A Tabela 4 mostra os resultados obtidos pela heurística e pelo método de Wang et al. [2013] nas instâncias G. A coluna 1 apresenta o nome da instância, a coluna 2 informa a quantidade de vértices da instância, a coluna 3 mostra o valor obtido pela heurística H_v, a coluna 4 o valor obtido por Wang et al. [2013], referenciado aqui por $GRASP_v$, e na coluna 5 mostramos o $GAP = \frac{GRASP_v - H_v}{GRASP_v} \times 100$, em porcentagem, entre os valores H_v e os valores $GRASP_v$.

Podemos verificar dos resultados apresentados na Tabela 4 que a nossa heurística alcançou o mesmo valor obtido pelo método GRASP nas instâncias G_{48}, G_{49} e G_{50} e nas instâncias restantes

o valor da nossa heurística ficou abaixo dos valores obtidos pelo método GRASP. Por outro lado, constatamos que os valores obtidos pela nossa heurística são aproximados, GAP menor que 1%, aos valores obtidos pelo método GRASP em 35 das 54 instâncias executadas, o que representa 64.8% das instâncias. Verifica-se que na média o GAP obtido foi de 1.59% para instâncias com 800 vértices, 0.58% para instâncias com 1000 vértices, 1.93% para instâncias com 2000 vértices, 0% para instâncias com 3000 vértices e 1.47% para todas as instâncias.

Tabela 3: Resultados de H_v nas instâncias G, com total de 5 execuções por instância.

Instância G	$\|V\|$	Melhor valor	σ	Média	Tempo (s)	Critério de parada
G_1	800	11590	9.31	11581	71.00	LE
G_2	800	11596	20.23	11578	64.74	LE
G_3	800	11611	11.44	11599	70.68	LE
G_4	800	11618	18.19	11592	63.23	LE
G_5	800	11626	16.51	11602	72.59	LE
G_6	800	2159	5.43	2151	92.49	LE
G_7	800	1963	5.90	1954	62.19	LE
G_8	800	1983	18.35	1963	59.20	LE
G_9	800	2037	22.75	2005	60.50	LE
G_{10}	800	1993	21.22	1977	59.74	LE
G_{11}	800	528	2.49	524	49.77	LE
G_{12}	800	524	4.10	518	51.58	LE
G_{13}	800	552	1.88	550	50.15	LE
G_{14}	800	3038	4.49	3032	63.97	LE
G_{15}	800	3024	4.64	3017	66.53	LE
G_{16}	800	3025	5.09	3020	57.56	LE
G_{17}	800	3017	4.49	3013	58.14	LE
G_{18}	800	972	7.13	964	67.13	LE
G_{19}	800	893	6.59	883	67.13	LE
G_{20}	800	927	0.94	925	57.77	LE
G_{21}	800	918	2.94	914	61.50	LE
G_{22}	2000	13291	23.31	13274	501.08	LE
G_{23}	2000	13295	36.3	13253	901.36	LE
G_{24}	2000	13297	27.15	13267	900.43	LE
G_{25}	2000	13299	37.2	13272	1800.43	TL
G_{26}	2000	13290	30.07	13243	1800.79	TL
G_{27}	2000	3300	9.89	3286	1800.67	TL
G_{28}	2000	3252	15.89	3236	1801.60	TL
G_{29}	2000	3361	11.58	3345	1800.75	TL
G_{30}	2000	3384	27.36	3350	1800.39	TL
G_{31}	2000	3251	1.00	3250	1800.06	TL
G_{32}	2000	1304	0.00	1304	1800.00	TL
G_{33}	2000	1280	3.00	1277	1800.13	TL
G_{34}	2000	1264	4.71	1260	1655.59	LE
G_{35}	2000	7589	0.47	7588	1800.75	TL
G_{36}	2000	7583	8.48	7577	1800.40	TL

G_{37}	2000	7608	9.89	7601	1802.02	TL
G_{38}	2000	7598	2.35	7596	1800.64	TL
G_{39}	2000	2345	1.88	2342	1801.59	TL
G_{40}	2000	2355	0.94	2354	1801.96	TL
G_{41}	2000	2322	4.71	2315	1801.15	TL
G_{42}	2000	2426	1.24	2424	1800.35	TL
G_{43}	1000	6628	3.77	6625	1205.23	LE
G_{44}	1000	6637	12.25	6619	1290.27	LE
G_{45}	1000	6641	13.67	6623	199.68	LE
G_{46}	1000	6621	4.24	6618	1310.77	LE
G_{47}	1000	6637	12.25	6628	1335.60	LE
G_{48}	3000	6000	0.00	6000	1802.91	TL
G_{49}	3000	6000	0.00	6000	1802.04	TL
G_{50}	3000	5880	20.78	5868	1800.46	TL
G_{51}	1000	3808	10.37	3800	1094.01	LE
G_{52}	1000	3820	9.89	3813	1083.17	LE
G_{53}	1000	3813	10.84	3805	1057.41	LE
G_{54}	1000	3815	12.25	3806	1058.19	LE

Tabela 4: Comparação dos resultados de H_v e $GRASP_v$ nas instâncias G.

Instância G	$\|V\|$	H_v	$GRASP_v$	$GAP(\%)$
G_1	800	11590	11624	0.29
G_2	800	11596	11620	0.21
G_3	800	11611	11620	0.08
G_4	800	11618	11646	0.24
G_5	800	11626	11631	0.04
G_6	800	2159	2178	0.87
G_7	800	1963	2006	2.14
G_8	800	1983	2005	1.10
G_9	800	2037	2054	0.83
G_{10}	800	1993	2000	0.35
G_{11}	800	528	564	6.38
G_{12}	800	524	556	5.76
G_{13}	800	552	582	5.15
G_{14}	800	3038	3063	0.82
G_{15}	800	3024	3050	0.85
G_{16}	800	3025	3052	0.88
G_{17}	800	3017	3047	0.98
G_{18}	800	972	992	2.02
G_{19}	800	893	906	1.43
G_{20}	800	927	941	1.49
G_{21}	800	918	931	1.40
G_{22}	2000	13291	13349	0.43
G_{23}	2000	13295	13332	0.28

G_{24}	2000	13297	13324	0.20
G_{25}	2000	13299	13326	0.20
G_{26}	2000	13290	13313	0.17
G_{27}	2000	3300	3325	0.75
G_{28}	2000	3252	3287	1.06
G_{29}	2000	3361	3394	0.97
G_{30}	2000	3384	3402	0.53
G_{31}	2000	3251	3299	1.45
G_{32}	2000	1304	1406	7.25
G_{33}	2000	1280	1374	6.84
G_{34}	2000	1264	1376	8.14
G_{35}	2000	7589	7661	0.94
G_{36}	2000	7583	7660	0.83
G_{37}	2000	7608	7670	0.81
G_{38}	2000	7598	7670	0.94
G_{39}	2000	2345	2397	2.17
G_{40}	2000	2355	2392	1.55
G_{41}	2000	2322	2398	3.17
G_{42}	2000	2426	2474	1.94
G_{43}	1000	6628	6660	0.48
G_{44}	1000	6637	6649	0.18
G_{45}	1000	6641	6654	0.20
G_{46}	1000	6621	6649	0.42
G_{47}	1000	6637	6656	0.29
G_{48}	3000	6000	6000	0.00
G_{49}	3000	6000	6000	0.00
G_{50}	3000	5880	5880	0.00
G_{51}	1000	3808	3847	1.01
G_{52}	1000	3820	3850	0.78
G_{53}	1000	3813	3848	0.91
G_{54}	1000	3815	3850	0.91

5. Conclusão

Nesse trabalho desenvolvemos uma heurística que combina probabilidade com uma busca local guiada para encontrar soluções para o problema do corte máximo. O desempenho da heurística foi analisado em comparação ao melhor método GRASP que também usa probabilidade em sua composição para resolver o corte máximo. Na análise, foi utilizada uma amostra de 54 instâncias da literatura com tamanhos variando entre 800 – 3000 vértices. Embora o algoritmo implementado da heurística tenha sido executado com limitações de tempo e número máximo de execuções, observamos nos experimentos realizado que aumentar o tempo limite de execução ou aumentar o número de iterações não resultou, de maneira significativa, em melhores soluções encontradas pela heurística.

Na comparação dos resultados, constatamos que os valores obtidos pela nossa heurística são próximos aos valores obtidos pelo método GRASP, com um GAP menor que 1% em 35 das 54 instâncias executadas. Na média o GAP da nossa heurística ficou 1.47% para todas as instâncias. Vale destacar que, na comparação foi levando em consideração o melhor valor obtido por cada

método, não sendo possível comparar nos experimentos o tempo de execução devido a configuração de máquinas distintas usadas em cada método.

Apesar dos resultados próximos ao encontrados pelo método GRASP, a heurística apresentada precisa de atualizações para que possa ser mais competitiva em termos de encontrar melhores soluções. Para trabalhos futuros, pretendemos testar formas diferentes de gerar uma boa solução inicial para a heurística, como por exemplo, algoritmos genéticos ou colônias de formigas. Além disso, outra atualização que pretendemos investigar é o uso combinado de desigualdades válidas para guiar a busca local e usar busca tabu para armazenar as melhores soluções aceitas pelas desigualdades válidas.

Referências

Arráiz, E. e Olivo, O. (2009). Competitive simulated annealing and tabu search algorithms for the max-cut problem. In *Proceedings of the 11th annual conference on genetic and evolutionary computation*, p. 1797–1798.

Barahona, F., Grötschel, M., Jünger, M., e Reinelt, G. (1988). An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513.

Benlic, U. e Hao, J.-K. (2013). Breakout local search for the max-cutproblem. *Engineering Applications of Artificial Intelligence*, 26(3):1162–1173.

Bouhmala, N. (2019). Combining simulated annealing with local search heuristic for max-sat. *Journal of Heuristics*, 25(1):47–69.

Boykov, Y. Y. e Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, volume 1, p. 105–112. IEEE.

Helmberg, C. e Rendl, F. (2000). A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673–696.

Karp, R. M. (1972). *Reducibility among combinatorial problems*. Springer.

Kim, Y.-H., Yoon, Y., e Geem, Z. W. (2019). A comparison study of harmony search and genetic algorithm for the max-cut problem. *Swarm and evolutionary computation*, 44:130–135.

Kochenberger, G. A., Hao, J.-K., Lü, Z., Wang, H., e Glover, F. (2013). Solving large scale max cut problems via tabu search. *Journal of Heuristics*, 19(4):565–571.

Krislock, N., Malick, J., e Rouoin, F. (2014). Improved semidefinite bounding procedure for solving max-cut problems to optimality. *Mathematical Programming*, 143(1–2):61–86.

Liers, F., Nieberg, T., e Pardella, G. (2011). Via minimization in vlsi chip design application of a planar max-cut algorithm.

Myklebust, T. G. (2015). Solving maximum cut problems by simulated annealing. *arXiv preprint arXiv:1505.03068*.

Soares, P. L. B. (2018). *Problemas quadráticos binários: abordagem teórica e computacional*. PhD thesis, Universidade Federal do Ceará, Fortaleza, CE.

Wang, Y., Lü, Z., Glover, F., e Hao, J.-K. (2013). Probabilistic grasp-tabu search algorithms for the ubqp problem. *Computers & Operations Research*, 40(12):3100–3107.