Uma abordagem paralela para resolução do MWPSP

Rafael S. Castro[1], **Humberto J. Longo**[1], **Wellington S. Martins**[1]

[1]Instituto de Informática – Universidade Federal de Goiás (UFG)
Caixa Postal 15.064 – 91.501-970 – Goiânia – GO – Brazil

rafaelcastrosilva@inf.ufg.br,{longo,wsmartins}@ufg.br

Abstract. *Algorithms for the problem of identifying a maximum edge-weight planar subgraph of a given edge-weighted graph G are relevant in a wide variety of application areas. In this paper, we propose a new local search constructive heuristic algorithm for this $\mathcal{NP}$-hard problem that is based on existing methods for the problem and uses a parallel approach on GPU.*

Resumo. *O problema de se identificar em um grafo G, ponderado nas arestas, um subgrafo planar de peso máximo pertence à classe $\mathcal{NP}$-difícil. Esse problema é importante na modelagem e resolução de problemas das mais diversas áreas. É proposto um novo algoritmo heurístico de busca local, baseado em métodos previamente existentes de construção, o qual usa paralelismo de GPU.*

1. Introdução

Seja um grafo finito, simples $G = (V(G), E(G), w)$, com conjunto de vértices $V(G)$, conjunto de arestas $E(G)$, pesos não negativos $w(e), e \in E(G)$ associados às arestas (w:$E(G) \rightarrow \mathbb{R}^+ \cup \{0\}$) e $|V(G)| = n$. O objetivo do problema do subgrafo planar maximal (*Maximum-weight Planar Subgraph Problem – MWPSP*) é encontrar um subgrafo gerador planar $G' = (V(G), E(G'))$ de G, com $E(G') \subseteq E(G)$ e $w(G') = \sum_{e \in E(G')} w(e)$ sendo o máximo possível. Uma instância do MWPSP é denotada por *MWPSP*(G, w) e se G' é um subgrafo de G, então $w(G')$ é chamado de *peso* de G'.

Vários algoritmos heurísticos existentes (ou simplesmente *heurísticas*) para o MWPSP, como o *PMFG* [Tumminello et al. 2005] e o *TMFG* [Massara et al. 2017], são baseados em processos iterativos de aplicação de operações topológicas simples. Em geral, tais heurísticas começam com um subgrafo K_4 (*tetraedro* ou semente inicial) de uma instância *MWPSP*(G, w) e progressivamente incrementam-a pela aplicação de operações topológicas, como a inserção na solução parcial corrente, a cada iteração, de um novo vértice e arestas incidentes a ele. Nesse caso, a escolha do novo vértice e das arestas é feita de acordo com o maior incremento possível na função objetiva.

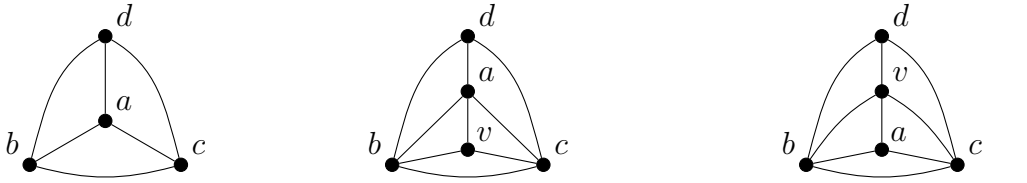
A heurística construtiva *All seeds* [Coelho et al. 2016] utiliza as operações *Face* e *Edge Dimpling* (veja a Seção 2) como sub-rotinas no processo de construção de soluções viáveis para o MWPSP. A heurística explora todas as possíveis sementes como entrada e separadamente aplica essas duas operações para a expansão das sementes. Em cada iteração, um novo vértice é inserido no subgrafo corrente, de acordo com o maior ganho local que uma das duas operações pode produzir. O melhor resultado obtido dentre todas as sementes testadas é então retornado como a solução final.

Este trabalho descreve uma versão paralela de granularidade fina da heurística *RestrictedSeeds*. Essa variante da *All seeds* considera apenas um subconjunto restrito de todas os possíveis tetraedros iniciais e tem complexidade assintótica bastante inferior à da *All*

Seeds. Nos testes realizados, com uso de GPU, não houve perda significativa na qualidade dos resultados obtidos quando comparados aos obtidos pela *All Seeds* (em média, inferior a 0,69% para as instâncias testadas). Na Seção 2 são explicados os movimentos topológicos utilizados na heurística. Na Seção 3 é definido o algoritmo desenvolvido nesse trabalho e como o paralelismo de GPU foi aplicado. Na Seção 4 são apresentados alguns resultados produzidos pela heurística, junto com o *speedup* obtido na versão paralelizada. Na Seção 5 é feita uma breve conclusão sobre o trabalho e sugestões para trabalhos futuros.

2. Operações topológicas

Operações topológicas (*local moves*) são técnicas fundamentais envolvendo vértices, arestas ou faces de um grafo planar que permitem transformar um dado grafo planar maximal em um novo grafo com as mesmas características. A operação construtiva *Face Dimpling*, originalmente descrita em [Foulds and Robinson 1978] é também denotada como *Vertex Insertion* [Leung 1992, Merker and Wäscher 1997] ou movimento T_2 [Massara et al. 2017]. A operação construtiva *Edge Dimpling* [Alexander 1930] é também chamada de operação A [Massara et al. 2017], em honra a Alexander.



(a) Grafo G original. **(b) Inserção do vértice v na face $\langle a, b, c\rangle$.** **(c) Inserção do vértice v na aresta $\{a, d\}$.**

Figura 1. Operações *Face Dimpling* (a → b) e *Edge Dimpling* (a → c).

Seja $G = (V(G), E(G))$ um grafo planar maximal, tal que o vértice $u \notin V(G)$ e xyz define uma face de G. A operação *Face Dimpling* cria um novo grafo planar maximal $G' = (V(G) \cup \{v\}, E(G) \cup \{vx, vy, vz\})$. A operação *Edge Dimpling* primeiramente remove uma aresta $ux \in E(G)$, comum às faces uwx, uxy de G. Em seguida, um novo vértice $v \notin V(G)$ é adicionado no interior da face $uwxy$ (resultante da remoção da aresta ux). Finalmente, o vértice v é conectado aos vértices u, w, x, y. Dessa forma, as faces originais uwx, uxy desaparecem e quatro novas faces uwv, uvy, vwx e vxy surgem. Essas duas operações topológicas são ilustradas na Figura 1.

3. Heurística *Restricted Seeds*

A construção do subconjunto $\mathcal{C}$ de *tetraedros* (passo 1) é a parte crítica do Algoritmo 1, pois soluções viáveis para o MWPSP são construídas com sucessivas aplicações da operação *face dimpling* ou da *edge dimpling* a partir das *sementes* em $\mathcal{C}$. Na rotina $FilteredSeeds(G, x, y, z)$ (passo 1), inicialmente são selecionadas apenas as y sementes mais pesadas. Dentre estas, as z ($z \leqslant y$) mais pesadas são adicionadas no subconjunto C. Depois, mais x sementes ($x + z \leqslant y$) são escolhidas aleatoriamente entre as $(y - z)$ sementes restantes e também adicionadas em C. Ao final da construção de $\mathcal{C}$, $|\mathcal{C}| = x + z$. Em seguida, a partir dessas sementes, subgrafos de G são construídos com a sucessiva aplicação das operações *face* e *edge dimpling* (passos 4–15).

Como a evolução de uma solução a partir de uma semente é independente das demais sementes, foi utilizado o processamento paralelizado em GPU na construção das

soluções. A abordagem foi separar a construção das soluções em diferentes *threads*, uma para cada semente, e retornar, ao final da execução de todas elas, a melhor solução obtida. É necessária a aplicação sucessiva de $n - 4$ operações topológicas (passos 8–15, $\mathcal{O}(1)$ cada um, exceto os passos 9 e 10). Os passos 9 e 10 determinam o melhor movimento e ambos têm complexidade $\mathcal{O}(n^2)$. Porém, pode-se também aplicar paralelismo nesses dois passos. Se a quantidade disponível de processadores, ou nesse caso *threads* em GPU, é da ordem de $\mathcal{O}(n^2)$, então essa tarefa pode ser resolvida em $\mathcal{O}(\log(n))$, usando-se algum algoritmo paralelo de encontrar o máximo em um subconjunto. Portanto, a complexidade final do algoritmo proposto é $\mathcal{O}((x + z)n \log(n))$. Contudo, se $x + z = \mathcal{O}(1)$, então a complexidade final da heurística proposta se mantém em apenas $\mathcal{O}(n \log(n))$, sem perda significativa na qualidade final da melhor solução encontrada em relação ao método proposto por [Coelho et al. 2016], conforme relatado na Seção 4.

Algoritmo 1: RestrictedSeeds(G, x, y, z)

Entrada: Grafo $G = (V(G), E(G))$ e parâmetros $x, y, z \in \mathbb{N}$ de redução do espaço de busca.
Saída: Grafo planar maximal G', filtrado de G.

1 $\mathcal{C} \leftarrow FilteredSeeds(G, x, y, z)$;
2 $MaxWeight \leftarrow 0$;
3 **para** $k \leftarrow 1$ **até** $|\mathcal{C}|$ **faça em paralelo**
4 $\quad \mathcal{C}_k \leftarrow \{v_1^k, v_2^k, v_3^k, v_4^k\} \in \mathcal{C}$;
5 $\quad \mathcal{S} \leftarrow V(G) - \{\mathcal{C}_k\}$;
6 $\quad \mathcal{E} \leftarrow \{\{v_1^k, v_2^k\}, \{v_1^k, v_3^k\}, \{v_1^k, v_4^k\}, \{v_2^k, v_3^k\}, \{v_2^k, v_4^k\}, \{v_3^k, v_4^k\}\}$;
7 $\quad \mathcal{F} \leftarrow \{\{v_1^k, v_2^k, v_3^k\}, \{v_1^k, v_2^k, v_4^k\}, \{v_1^k, v_3^k, v_4^k\}, \{v_2^k, v_3^k, v_4^k\}\}$;
8 $\quad$ **enquanto** $(\mathcal{S} \neq \varnothing)$ **faça**
9 $\quad\quad face_move \leftarrow \{$maior ganho com $\{s, f\}$, $\forall f \in \mathcal{F}$ e $s \in \mathcal{S}\}$;
10 $\quad\quad edge_move \leftarrow \{$maior ganho com $\{s, e\}$, $\forall e \in \mathcal{E}$ e $s \in \mathcal{S}\}$;
11 $\quad\quad$ **se** $(face\_move \geqslant edge\_move)$ **então**
12 $\quad\quad\quad$ Chame a rotina de *Face Dimpling*;
13 $\quad\quad$ **senão**
14 $\quad\quad\quad$ Chame a rotina de *Edge Dimpling*;
15 $\quad\quad$ Atualize os conjuntos $\mathcal{F}$, $\mathcal{E}$ e $\mathcal{S}$;
16 $\quad \mathcal{R} \leftarrow \mathcal{E} \cup \mathcal{R}$; // $\mathcal{R}$: resultados das execuções paralelas.
17 $\mathcal{M}' \leftarrow \max(\mathcal{R})$;
18 **retorna** $G' = (V, \mathcal{M}')$.

4. Resultados

Os testes com a versão sequencial da *Restricted Seeds* ocorreram em computador com 16Gb de RAM, processador Intel(R) Core(TM) i5 CPU 760 @ 2.80GHz e SO Ubuntu 18.04. As versões paralelas da *Restricted Seeds* e da *All Seeds* foram testadas utilizando-se CPU Intel(R) Xeon(R) @ 2.30GHz modelo 63 e GPU Tesla T4 2560 CUDA Cores, ambos dispositivos com 16GB de memória principal.

A Tabela 1 compara os resultados obtidos com as heurísticas *All Seeds* [Coelho et al. 2016] e *Restricted Seeds*. A coluna rotulada $\boldsymbol{n}$ contém as quantidades de vértices das instâncias. As colunas **AS-GPU** e **AS-Val.** contêm os tempos de processamento na versão paralelizada e o peso do grafo planar, respectivamente, obtidos pela *All Seeds*. As colunas **RS-Seq.**, **RS-GPU** e **RS-Val.** apresentam os tempos consumidos pela

Restricted Seeds, nas versões sequencial e paralela, e o peso do grafo planar obtido, respectivamente. A coluna **gap(%)** mostra a diferença percentual entre os valores obtidos pelas duas heurísticas (em média, inferior a 0,69%). A última coluna contém o speedup da versão paralela da heurística *Restricted Seeds* em relação à sua versão sequencial.

Tabela 1. Resultados obtidos pelas heurísticas *Restricted Seeds* e *All Seeds*.

n	AS-GPU	AS-Val.	*RS*-Seq.	*RS*-GPU	*RS*-Val.	gap (%)	*speedup*
30	0,352	13644	4,689	0,077	13586	0,43	60,89
40	1,459	19454	8,603	0,150	19377	0,40	57,35
50	8,402	25126	14,215	0,306	24904	0,88	46,45
60	22,613	30432	20,969	0,356	30210	0,73	58,90
70	74,502	36410	28,098	0,428	36208	0,55	65,64
80	189,905	41771	33,631	0,587	41443	0,79	57,29
90	483,651	47301	47,836	0,837	46933	0,78	57,15
100	1.034,420	53377	58,208	1,025	52894	0,90	56,78

5. Conclusão e trabalhos futuros

Este trabalho apresentou um novo algoritmo heurístico para a resolução do MWPSP, o qual constrói soluções (subgrafos planares maximais) a partir de um subgrafo planar básico e de pequenos incrementos sucessivos na estrutura do subgrafo corrente. Em comparação à *All-Seeds* [Coelho et al. 2016], este novo algoritmo possui complexidade assintótica bastante inferior. Além disso, foi mostrado como a implementação paralela do método pode trazer um ganho de tempo ainda mais substancial.

Como sugestão de novos trabalhos sugere-se a adição de novos movimentos topológicos na construção de soluções, além dos já citados aqui. Outra possibilidade seria mudar o critério de seleção do movimento topológico a ser usado. Neste trabalho o critério usado foi o do maior incremento local (passos 9 e 10). Uma opção seria, por exemplo, escolher de forma aleatória entre um certo espaço de possíveis movimentos com maiores incrementos, semelhante a estratégia utilizada na filtragem das sementes (passo 1).

Referências

Alexander, J. W. (1930). The combinatorial theory of complexes. *Ann Math*, 31:292–320.

Coelho, V. S., Martins, W. S., Foulds, L. R., Dias, E. S., Castonguay, D., and Longo, H. J. (2016). Uma proposta de solução aproximada para o problema do subgrafo planar de peso máximo. In *XVII Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD 2016)*, pages 16–27.

Foulds, L. R. and Robinson, D. F. (1978). Graph theoretic heuristics for the plant layout problem. *Int J Prod Res*, 16(1):27–37.

Leung, J. (1992). A New Graph-Theoretic Heuristic for Facility Layout. *Manage Sci*, 38(4):594–605.

Massara, G. P., Di Matteo, T., and Aste, T. (2017). Network filtering for big data: Triangulated maximally filtered graph. *J Complex Netw*, 5(2):1–18.

Merker, J. and Wäscher, G. (1997). Two new heuristic algorithms for the maximal planar layout problem. *OR Spectr*, 19(2):131–137.

Tumminello, M., Aste, T., Di Matteo, T., and Mantegna, R. N. (2005). A tool for filtering information in complex systems. *Proc Natl Acad Sci*, 102(30):10421–10426.