XXXIII Congresso de Iniciação Científica da Unicamp

Algoritmos para o Problema do Conjunto Dominante Justo *

Orientando: Leonardo Valente Nascimento - IC/UNICAMP
Orientador: Lehilton Lelis Chaves Pedrosa - IC/UNICAMP

Resumo

No Problema do Conjunto Dominante (DS), dado um grafo $G = (V, E)$, o objetivo é encontrar um subconjunto $D \subseteq V$ de tamanho mínimo tal que $N[D] = V$.

Consideramos o Problema do Conjunto Dominante β-Justo (β-FDS), uma variante de DS em que V é a união disjunta de vértices vermelhos e azuis, cada vértice deve ser atribuído a um vértice dominante em D, e a razão entre o número de vértices de cada cor atribuídos a um elemento de D deve ser pelo menos β.

Esse problema possui aplicações em agrupamento justo (*fair clustering*), que busca evitar a sub-representação de grupos, geralmente presente em soluções produzidas por algoritmos de agrupamento padrão.

Mostramos que β-FDS é $W[1]$-difícil quando parametrizado por $k + tw$ para cada $\beta > 0$, onde k é o tamanho da solução e tw é a treewidth do grafo. Quando consideramos que $\beta = \frac{p}{q}$ para inteiros p e q, apresentamos um algoritmo $\mathcal{O}(nq\Delta)$ para árvores e um algoritmo $\mathcal{O}(3^{tw}(q\Delta)^{2tw+2}n)$ para grafos gerais que resolve β-FDS, onde Δ é o grau máximo do grafo de entrada.

Palavras-Chave: Teoria dos grafos, Algoritmos Parametrizados, Clusterização

1 Introdução

No problema do Conjunto Dominante (DS), dado um grafo $G = (V, E)$ e um inteiro k, devemos dizer se existe um conjunto $D \subseteq V$ tal que $N[D] = V$ (ou seja, todo vértice de V está em D ou é vizinho de D) e $|D| \leq k$. Esse problema é extremamente importante tanto de um ponto de vista teórico — dado que é um problema NP-Completo [Garey and Johnson, 1979] muito estudado — quanto do ponto de vista prático — dadas as aplicações que DS e suas variantes têm em diversos problemas.

Uma das aplicações de Conjunto Dominante é como ferramenta de *clusterização* (ou agrupamento) [Haenn et al., 2024] [Krishnam and Varma, 2011]. Problemas de agrupamento consistem em particionar um conjunto de dados em grupos, de tal forma que os elementos de cada grupo são "similares" de acordo com alguma métrica. Esse tipo de problema também possui diversas aplicações [Ghosal et al., 2019]. DS é útil em clusterização pois ao considerar um grafo $G = (V, E)$ onde V corresponde aos elementos do conjunto de dados e E corresponde a

*O presente trabalho foi realizado com apoio do CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico, projetos número 312271/2023-9, 404315/2023-2, 124332/2024-2, e FAEPEX/Unicamp, projeto número 2422/23.

pares de elementos "similares", cada elemento do conjunto dominante de G pode ser visto como o *centro* de um cluster [Hochbaum and Shmoys, 1985].

Problemas de clusterização clássicos não impõem restrições sobre o tamanho ou distribuição dos pontos de um cluster, o que pode levar a sub-representação ou viés contra grupos de elementos em certas aplicações [Chierichetti et al., 2017]. Tome como exemplo ilustrativo uma aplicação onde queremos agrupar notícias de caráter parecido. Um algoritmo de clusterização clássico pode produzir *clusters* com uma quantidade exagerada de notícias de certo cunho político, por exemplo, o que pode ser indesejado. Isso motiva a área de *fair clustering*, onde buscamos algoritmos de clusterização que evitem esse tipo de viés.

Nesse trabalho, aplicamos o conceito de "balance" introduzido em [Chierichetti et al., 2017] ao problema do Conjunto Dominante, criando a variante Conjunto Dominante β-Justo (β-FDS). Em β-FDS, passamos a considerar um grafo $G = (V, E)$ onde os vértices são coloridos com *azul* ou *vermelho*, e queremos decidir se existe um conjunto dominante de G de tamanho k tal que cada *cluster* induzido pelo conjunto dominante tenha uma razão de vértices azuis e vermelhos de pelo menos β (o problema será definido formalmente em 2).

Em particular, estudamos a complexidade parametrizada de β-FDS. Mostramos que β-FDS é W[1]-difícil quando parametrizado por $k + tw$, onde k é o tamanho do conjunto dominante e tw é a largura arbórea do grafo de entrada indicando a possível inexistência de algoritmos eficientes nesses parâmetros.

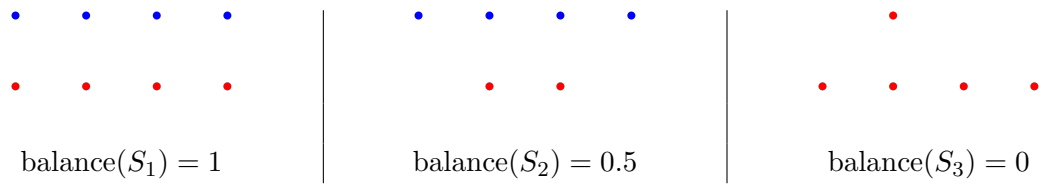
Também obtemos resultados positivos na forma de algoritmos. Por conveniência, vamos dizer que $\beta = \frac{p}{q} > 0$ para inteiros p e q. Para árvores, temos um algoritmo $\mathcal{O}(nq\Delta)$ para β-FDS, onde Δ é o grau máximo da árvore de entrada. Para grafos gerais, temos um algoritmo $\mathcal{O}(3^{tw}(q\Delta)^{2tw+2}n)$ para β-FDS, onde tw é a largura arbórea de G e Δ é o grau máximo do grafo de G.

2 Métodos e Definições

Dado um grafo $G = (V, E)$ e uma bipartição dos vértices em duas cores R e B, isto é, $V = R \cup B$, o *balanceamento* de um conjunto não-vazio $S \subseteq V$ é definido como

$$\text{balance}(S) = \frac{\min\{\,|S \cap R|, |S \cap B|\,\}}{\max\{\,|S \cap R|, |S \cap B|\,\}}.$$

O balanceamento captura a proporção da representação das cores em um *cluster*. Seguem exemplos de valores de balance de três conjuntos, a fim de melhor entendimento.



Em particular balance$(S_2) = 0.5$ pois as cores estão em uma proporção de 1 para 2. Aplicando a fórmula da definição, temos balance$(S_2) = \frac{\min\{4,2\}}{\max\{4,2\}} = \frac{2}{4} = 0.5$

Dizemos que um conjunto $D \subseteq V$ é conjunto dominante *β-justo* se existir um mapeamento $\alpha : V \to D$ tal que balance$(\alpha^{-1}(d)) \geq \beta$ para todo $d \in D$, isto é, o balanceamento do *cluster* associado a d é pelo menos β. Para alguma constante β, com $0 < \beta \leq 1$, o problema do Conjunto Dominante β-Justo (β-*Fair Dominating Set*, β-FDS) consiste em, dado um grafo

$G = (V, E)$, cujos vértices têm cores R ou B, e um inteiro k, decidir se existe conjunto dominante β-justo de tamanho no máximo k.

Também dizemos que um algoritmo é FPT em relação ao parâmetro k quando sua complexidade de tempo é da forma $f(k)n^{\mathcal{O}(1)}$ para alguma função computável f. Ademais, um problema é FPT quando possui algoritmo FPT que o resolva.

Para aqueles mais familiares com complexidade clássica, a classe P corresponderia a classe FPT em complexidade parametrizada, enquanto a classe W[1] corresponderia a classe NP. Sendo assim, acredita-se que se um problema é W[1]-difícil, não há algoritmo FPT que resolva ele para aqueles parâmetros. Definições formais da classe W[1] se encontram em [Cygan et al., 2015].

Será omitida também a definição de largura arbórea (treewidth), mas a mesma se encontra em [Cygan et al., 2015]. Intuitivamente, a largura arbórea de um grafo representa o quão próximo ele está de uma árvore. Para algoritmos baseados em treewidth, assumimos que estamos equipados com uma decomposição em árvore de largura mínima do grafo de entrada para fins de simplicidade, novamente seguindo [Cygan et al., 2015].

Para as complexidades de tempo dos algoritmos apresentados, denotamos $n = |V|$ e $m = |E|$.

Por se tratar de um trabalho de teoria da computação, a obtenção dos resultados foi dada por meio de demonstrações matemáticas. Essas demonstrações foram verificadas pelo orientador durante reuniões realizadas ao longo do período de pesquisa. Quando pertinente, serão apresentadas ideias das demonstrações dos teoremas.

3 Algoritmos para β-FDS

Primeiramente, vale comentar que apesar de todo grafo possuir algum conjunto dominante (em particular, V é conjunto dominante de qualquer grafo), nem todo grafo admite conjunto dominante β-justo. Por exemplo, não há conjunto dominante 1-justo para grafos com uma quantidade diferente de vértices de cada cor. Entretanto, ao menos para $\beta = 1$, é fácil checar se existe conjunto dominante β-justo.

Teorema 1. *Existe algoritmo que decide se um grafo $G = (V, E)$ com coloração $\chi : V \to \{R, B\}$ admite conjunto dominante 1-justo em tempo $\mathcal{O}(n^{5/2} + m)$.*

Note que o algoritmo do teorema 1 não resolve 1-FDS, pois não garante que o conjunto dominante tem tamanho no máximo k. Em seguida, mostramos que o problema é FPT quando parametrizado por $tw+\Delta$, sendo Δ o maior grau entre os vértices do grafo. Para isso, mostramos um lema intermediário:

Lema 2. *Seja $\beta = \frac{p}{q}$ para inteiros p e q. S é um conjunto β-justo se e somente se existe função $g : S \to [p, q] \cap \mathbb{Z}$ tal que:*

$$\sum_{r \in S \cap R} g(r) = \sum_{b \in S \cap B} g(b) \iff \sum_{r \in S \cap R} g(r) - \sum_{b \in S \cap B} g(b) = 0$$

É possível demonstrar o lema 2 por meio de um argumento de continuidade discreta. Esse lema é extremamente útil pois traz uma caracterização muito mais fácil de trabalhar para β-FDS, possibilitando o uso de programação dinâmica para resolver o problema. A partir disso, obtemos o seguinte teorema:

Teorema 3. *Quando $\beta = \frac{p}{q}$, existe algoritmo que resolve β-FDS em tempo $\mathcal{O}(3^{tw}(q\Delta)^{2tw+2}n)$ para grafos de grau no máximo Δ quando equipado com uma decomposição em árvore de largura tw.*

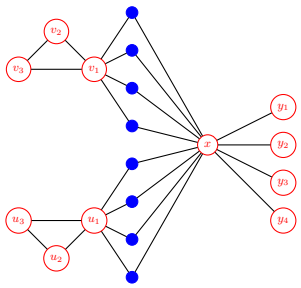


Figura 1: Redução de CDS para β-FDS.

O teorema 3 parte de uma programação dinâmica na decomposição em árvore do grafo de entrada. Em particular, utilizamos o conceito de *nice tree decomposition* com *introduce edge node*, assim como apresentado em [Cygan et al., 2015]. Sendo assim, precisamos apenas garantir que a condição do lema 2 valha. Uma análise cuidadosa das transições da programação dinâmica nos levam à complexidade enunciada.

Note que o teorema 3 indica que β-FDS é FPT em $\Delta + tw$, já que estamos trabalhando com β constante.

Teorema 4. *Existe algoritmo que resolve β-FDS para árvores em tempo $\mathcal{O}(nq\Delta)$.*

O algoritmo considera, para cada vértice v, um subproblema de encontrar um melhor *cluster* $\alpha^{-1}(v)$ tal que a condição do lema 2 valha. No final, reduzimos o problema para n instâncias de uma versão modificada de Bounded-Knapsack. Cada instância é resolvida em tempo $\mathcal{O}(\delta^2(v)q)$ a partir de pequenas modificações no algoritmo apresentado em [Kellerer et al., 2004], onde $\delta(v)$ corresponde ao grau do vértice v. Somando o tempo de todas as instâncias, temos $\mathcal{O}(\sum_{v\in V} \delta^2(v)q) \leq \mathcal{O}(q\Delta \sum_{v\in V} \delta(v)) = \mathcal{O}(nq\Delta)$.

4 W[1]-dificuldade de β-FDS

Para estudar a dificuldade de β-FDS, fazemos uma redução do Problema do Conjunto Dominante Capacitado (CDS). Dado um grafo $G = (V, E)$ e uma função de capacidade $c : V \to \mathbb{N}$, um subconjunto de vértices $D \subseteq V$ é um conjunto dominante capacitado se existir um mapeamento $\alpha : V \to D$ tal que $|\alpha^{-1}(d)| \leq c(d)$ para todo $d \in D$. O problema CDS consiste em, dado um grafo, capacidades nos vértices e um inteiro k, decidir se existe um conjunto dominante capacitado de tamanho no máximo k.

Sabemos que CDS é $W[1]$-difícil quando parametrizado por $k + tw$, em que tw é a largura arbórea do grafo em questão e k é o tamanho da solução [Dom et al., 2008][†]. Em seguida, iremos demonstrar que β-FDS também é $W[1]$-difícil para cada β racional via uma redução parametrizada.

Teorema 5. *Para cada β racional tal que $0 < \beta \leq 1$, β-FDS é $W[1]$-difícil quando parametrizado por $k + tw$, sendo tw a largura arbórea do grafo de entrada e k o tamanho da solução.*

Ideia da demonstração. Como β é racional entre 0 e 1, podemos escrevê-lo como uma fração irredutível $\frac{p}{q}$, onde p e q são constantes inteiras positivas e $p \leq q$. Dada uma instância (G, c, k)

[†]Em [Dom et al., 2008], um conjunto dominante capacitado é definido em termos de um mapeamento $\alpha : (V \setminus D) \to D$, mas é conveniente considerar um mapeamento $\alpha : V \to D$ no contexto de particionamento justo e é possível demonstrar que ambas as variantes são $W[1]$-difíceis modificando ligeiramente a redução apresentada no artigo original.

de CDS, em que $G = (V, E)$ é um grafo de largura arbórea tw, vamos construir uma instância (G', R, B, k') de β-FDS em que $G' = (V', E')$ é um grafo com largura arbórea no máximo $tw' = q \cdot tw + 1$ e $k' = k + 1$ (veja Figura 1).

Para cada vértice $v \in V$, crie uma clique com q vértices em R (vermelhos) denotados por $v_1, v_2, \ldots, v_q$. Além disso, para cada aresta $(u, v) \in E$, adicione uma aresta entre u_i e v_j para todo $1 \leq i, j \leq q$. Observe que, nesse momento, os vértices de cada clique são vizinhos gêmeos e G' é igual ao produto entre G e K_q.

Agora, para cada vértice $v \in V$, crie $p \cdot c(v)$ vértices em B (azuis) denotados por $v'_1, v'_2, \ldots, v'_{p \cdot c(v)}$ e ligue cada vértice v'_i ao vértice v_1, que é a primeira cópia do vértice v na clique. Defina $C = \sum_{v \in V} c(v)$ e observe que o número de vértices azuis criados é $p \cdot C$.

Crie um vértice x vermelho e ligue esse vértice a cada um dos vértices azuis, isto é, cada vértice v'_i para $v \in V$ e $1 \leq i \leq p \cdot c(v)$. Finalmente, crie $q \cdot (C - |V|) - 1$ vértices vermelhos, denotados por $y_1, y_2, \ldots, y_{q \cdot (C - |V|) - 1}$, e ligue cada um deles a x. Isso completa a construção de G'.

Para completar a demonstração, é possível mostrar que cada solução para a instância de CDS corresponde a uma solução de β-FDS e vice-versa. A observação principal é que o número de vértices vermelhos é exatamente $q \cdot C$ enquanto o número de vértices azuis é exatamente $p \cdot C$. Isso implica que, para um conjunto dominante β-justo D com mapeamento α, temos balance$(\alpha^{-1}(d)) = \beta$ para cada vértice $d \in D$. Segue que cada *cluster* tem um múltiplo de q vértices vermelhos e um múltiplo de p vértices azuis. Além disso, é possível verificar que existe uma solução em que o número de vértices de G associados a cada d é no máximo $c(d)$. □

Referências Bibliográficas

[Chierichetti et al., 2017] Chierichetti, F., Kumar, R., Lattanzi, S., and Vassilvitskii, S. (2017). Fair clustering through fairlets. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. 2

[Cygan et al., 2015] Cygan, M., Fomin, F., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., and Saurabh, S. (2015). *Parameterized Algorithms*. Springer. 3, 4

[Dom et al., 2008] Dom, M., Lokshtanov, D., Saurabh, S., and Villanger, Y. (2008). Capacitated domination and covering: a parameterized perspective. In *Proceedings of the 3rd International Conference on Parameterized and Exact Computation*, IWPEC'08, page 78–90, Berlin, Heidelberg. Springer-Verlag. 4

[Garey and Johnson, 1979] Garey, M. and Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-completeness*. Books in mathematical series. W. H. Freeman. 1

[Ghosal et al., 2019] Ghosal, A., Nandy, A., Das, A. K., Goswami, S., and Panday, M. (2019). *A Short Review on Different Clustering Techniques and Their Applications*, pages 69–83. Springer Singapore. 1

[Haenn et al., 2024] Haenn, Q., Chardin, B., and Baron, M. (2024). *Clustering Under Radius Constraints Using Minimum Dominating Sets*, pages 14–23. Springer Nature Switzerland. 1

[Hochbaum and Shmoys, 1985] Hochbaum, D. and Shmoys, D. (1985). A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10:180–184. 2

[Kellerer et al., 2004] Kellerer, H., Pferschy, U., and Pisinger, D. (2004). *The Bounded Knapsack Problem*, pages 185–209. Springer Berlin Heidelberg. 4

[Krishnam and Varma, 2011] Krishnam, R. and Varma, S. (2011). Dominating sets and spanning tree based clustering algorithms for mobile ad hoc networks. *International Journal of Advanced Computer Science and Applications*, 2(2). 1