A simpler and faster algorithm to compute Induced Star Partitions in graphs

Guilherme C. M. Gomes[1], **Matheus T. Prates**[1]

[1] Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG) – Belo Horizonte, MG – Brazil

{gcm.gomes,matheus.prates}@dcc.ufmg.br

Abstract. *An induced star partition of a graph is a partition of its vertex set in which all parts must induce a star, i.e., each part i must induce a subgraph isomorphic to a K_{1,t_i} for some $t_i \geq 1$. We present a new algorithm for computing one such partition, which improves on previous results in the literature, being both simpler and more efficient.*

1. Introduction

This work focuses on the INDUCED STAR PARTITION problem, first introduced by [Saito and Watanbe 1993] and [Kelmans 1997]. Let $G = (V, E)$ be a connected undirected graph. Formally, an **induced star partition** of G is a partition $\{V_1, ..., V_k\}$ of $V(G)$ wherein each V_i is a *star*, i.e., $G[V_i]$ is isomorphic to a complete bipartite graph K_{1,t_i} for some $t_i \geq 1$. The aforementioned papers prove that such a partition exists if and only if the graph is not an *odd block graph*, i.e., a graph whose blocks are all odd-sized cliques. The *blocks* of a graph are its maximal biconnected subgraphs.

Other variants of the problem have also been studied in the literature. Namely, [Babenko and Gusakov 2013] reduced the non-induced version of STAR PARTITION to network flows, thus developing a $O(\sqrt{V}E)$ solution. Furthermore, [Shalu et al. 2022] proved that determining whether a graph has an induced star partition with at most k parts is NP-complete for several graph classes. Later, [Divya and Vijayakumar 2024] offered additional results on split graphs. Nevertheless, it is worth noting that the definition of an induced star partition in those two papers is slightly different from ours. In particular, they consider a single vertex to be a $K_{1,0}$ star, and therefore a partition of size $|V|$ can be constructed for any graph. In contrast, our work follows the definitions of [Kelmans 1997], where both sides of the bipartite graph must be non-empty.

From [Saito and Watanbe 1993], one can derive an $O(VE)$ algorithm for finding an induced star partition. [Kelmans 1997] also proposed a solution, though it is challenging in both implementation and analysis. Our study presents a simpler approach to the problem by exploring its ties to maximum matchings, which have been broadly used in algorithmic graph theory [Lovász and Plummer 2009]. Our method has a time complexity of $O(\sqrt{V}E)$ for general graphs and $O(V + E)$ for graphs with bounded maximum block size.

2. Our results

Our algorithm for finding an induced star partition works as follows. For each block of the graph, we compute its maximum matching using any of the well-known methods [Edmonds 1965, Micali and Vazirani 1980, Gabow and Tarjan 1989, Blum 1990]. Next, in linear time, we modify each of the matchings independently and then combine them to obtain an induced star partition. Note that articulation points are shared between blocks. For general graphs, the overall time complexity is $O(\sqrt{V}E)$, assuming maximum matching algorithm of [Micali and Vazirani 1980] is used. Note that this is the only superlinear step in our procedure. This implies that, for graphs in which the maximum block size is bounded by a constant, or when matchings of the blocks can be found in linear time, the whole algorithm is linear. We formalize this in Theorem 1.

Theorem 1. *Let $G = (V, E)$ be a graph and $\{B_1, ..., B_k\}$ its set of blocks. Assume that G is not an odd block graph. For every $i \in [k]$, let M_i be a maximum matching of $G[B_i]$. Given G and the set of matchings $\{M_1, ..., M_k\}$ as input, there is an $O(V + E)$ algorithm for computing an induced star partition of G.*

The remainder of this section provides a proof outline of Theorem 1. As G is not an odd block graph, some of its blocks are not odd cliques. For each such block, say B_i, we compute an induced star partition of $G[B_i]$. For convenience, let $B = B_i$ and $M = M_i$. We build an induced star partition of B with size $|M|$.

Initially, we place each edge $xy \in M$ in its own part P_{xy}. Now, consider a vertex $v \notin V(M)$. We know that $N(v) \subseteq V(M)$, by the maximality of M. Let $x \in N(v)$ such that $xy \in M$, but $y \notin N(v)$. Such a vertex may not exist, but assume that it does. We place v in part P_{xy} and claim that applying this construction iteratively yields an induced star partition of B. To prove this, let $xy \in M$ be an arbitrary edge. Seeing that M is maximum, $P_{xy} \backslash \{x, y\}$ must be an independent set. Moreover, there cannot be $a, b \in P_{xy}$ such that $xa, yb \in E(B)$; otherwise, we could find a larger matching by exchanging xy with xa and yb. Finally, our construction guarantees that no $v \in P_{xy}$ is adjacent to x, y both. This is sufficient to prove that the subgraph induced by P_{xy} is a star.

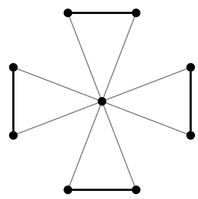

Figure 1. A $\mathrm{Wd}(3, n)$-windmill. The thick edges belong to the matching M.

In our construction, we assumed that, for all $v \notin V(M)$, M does not contain a perfect matching of $N(v)$. Suppose that is not the case, i.e., there is at least one vertex for which this rule does not apply, as depicted in Fig. 1. This structure is known as a Friendship Graph, or a $\mathrm{Wd}(3, n)$-windmill [Gallian 2022]. Our task is to find an alternative maximum matching M' without any such windmills. This can be done in linear time, with the help of the following lemmas.

Lemma 1 (Windmill Propagation)**.** *Let v be the center of a windmill w.r.t. M, and consider an edge $uw \in M$ such that $vu, vw \in E(B)$. We obtain a new maximum matching M' by taking M and replacing uw with vu. Then, (I) v is not the center of a windmill w.r.t. M'. Furthermore, suppose there exists an $x \in V(B)$, $x \neq w$, such that x is not the center of any windmill w.r.t. M. In this case, (II) x cannot be the center of a windmill w.r.t. M'.*

Proof. To prove (I), observe that, since $vu \in M'$, v is not the center of a windmill w.r.t. M'. Regarding (II), let $x \in V(B)$ be such that x is not the center of a windmill w.r.t M, but is one w.r.t. M'. This vertex x must be adjacent to both v and u, and cannot belong to $V(M')$. However, notice that $x \in V(M)$; otherwise, we could have added vx to M, contradicting its maximality. Because $V(M)\backslash V(M') = \{w\}$, no vertex $x \neq w$ exists. □

Lemma 2 (Windmill Isolation). *Let x, y be the centers of two distinct windmills w.r.t. M. It must hold that the distance between them, in edges, is at least 3.*

Proof. We know that $dist(x, y) > 1$, since M is maximal. Now, assume $dist(x, y) = 2$. Let $z \in V(B)$ such that $xz, yz \in E(B)$. Since x, y are both centers of a windmill, it must hold that $zr \in M$ for some $r \in V(B)$. Moreover, $xr, yr \in E(B)$. By swapping zr with xz and yr, we obtain a larger matching; a contradiction. □

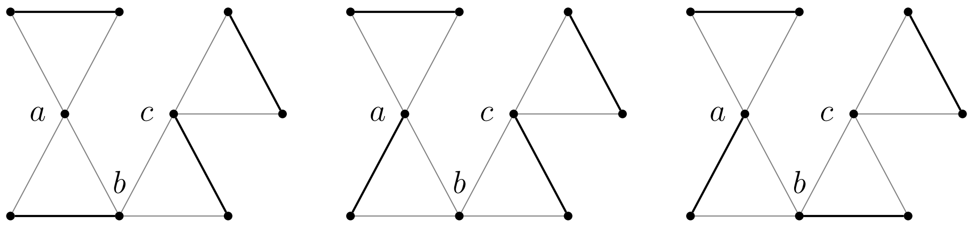

Figure 2. Propagation of a windmill along the path (a, b, c).

From Lemma 1, we deduce that we can propagate a windmill along a path, as shown in Fig. 2. After exchanging an edge and eliminating the windmill centered at the start of the path, we either reduce the total number of windmills in the graph, or we create a single new windmill centered at the next vertex in the path. With this in mind, let C be the set of vertices which are the center of a windmill at any given time. Assume that initially $|C| \geq 2$. Consider a spanning tree of B, which can be obtained in linear time [Kruskal 1956]. First, we root the tree at a vertex $v \in C$. Next, we process each of the vertices one by one, in order of depth, starting with the leaves. In each step, if the given vertex belongs to C, we either remove the windmill or propagate it to the immediate ancestor in the tree. We continue with this process until we reach the root. From Lemma 2, we can conclude that, at this point, the only windmill that remains is centered at the root.

We can now assume that $C = \{v\}$ for some $v \in V(B)$. Let $d(v) > 2$ be the degree of v in B; if $d(v) = 2$, then, given that B is not a triangle, we can propagate the windmill to a neighbor u with $d(u) > 2$. Without loss of generality, let $W = \{a, b, c, d, v\} \subseteq N[v]$ with $ab, cd \in M$. We omit the proof of the following lemma for brevity.

Lemma 3. *Suppose that $B[W]$ is not a clique and that $\{ab, cd\}$ is not an induced matching. Then, the windmill can be eliminated by exchanging edges within $B[W]$ alone.*

We now consider the following cases independently: (1) $d(v) < n - 1$ and (2) $d(v) = n - 1$. First, suppose (1). Let $x \in V(B)$ such that $dist(x, v) = 2$ and $xb \in E(B)$. We aim to assume there is a path P from x to d that does not pass through v, b, c, or a (see Fig.3). Suppose the contrary, i.e., such a path P does not exist for our chosen W. Because B is biconnected, $\{ab, cd\}$ is not an induced matching. Lemma 3 then allows us to assume $B[W] \cong K_5$. Additionally, there must be a path P' from a to x that avoids b. If $vy \in P'$, we find a suitable W by changing our choice of cd to include y. Conversely, if $v \notin P'$, we alter the matching M: $\{ab, cd\}$ is replaced with $\{ac, bd\}$, and we modify the labels accordingly. Thus, we can assume a path P exists as we defined.

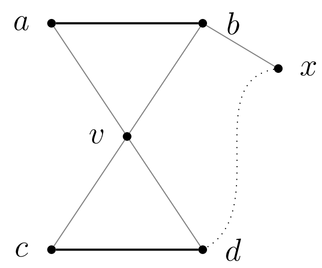

Figure 3. A windmill centered on v. The dotted line is a path from d to x.

From Lemma 1, we can propagate the windmill along P. If it reaches x, we deduce that both edges ax and bx must exist. From there, we proceed as shown in Fig. 4. Case (2) is similar: since B is not an odd clique, we assume there exists a W such that $B[W]$ is not a clique. Then, we show that there is a path connecting two non-adjacent vertices of W, and propagate the windmill along this path so that it disappears.

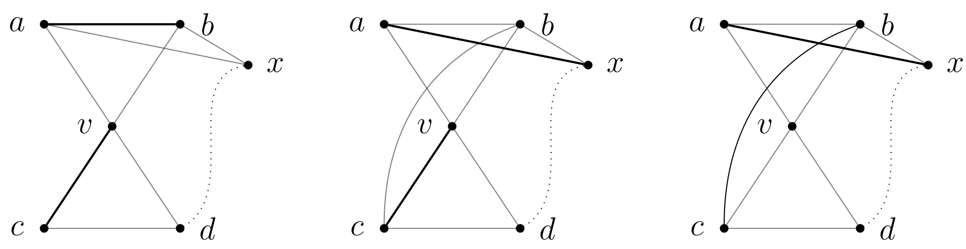

Figure 4. A sequence of operations to make the windmill from x disappear.

Lastly, consider the block-cut tree decomposition of G [Manber 1998]. Root it at one of the blocks which is not an odd clique. Then, for each block that is an odd clique, remove the edges from the articulation point connecting to its immediate ancestor. Finally, we process each block in decreasing order of depth. At each step, the partition of the current block might conflict with that of the children at the articulation points. This leads to a simple case analysis, allowing us to successfully merge the partitions.

3. Future Work

INDUCED STAR PARTITION is a specific case of a more general problem of interest for us. Let $Q, K \in \mathbb{N}$, $G = (V, E)$ be a connected graph, and $\phi : V \to [K]$ be a K-coloring of the vertices such that there is no connected monochromatic subgraph of size at least Q. Let us define the following operation: choose a vertex v and set $\phi(v)$ to a new value from $[K] \setminus \phi(v)$. If after the operation there is a maximal connected monochromatic subgraph of size at least Q, we must immediately remove it from the graph. The decision version of the problem asks whether it is possible to make the graph empty by using this operation any number of times. Suppose that $Q = 2$ and that there is a color $c \in [K]$ such that $\forall v : \phi(v) \neq c$. In this scenario, determining a sequence of operations to make the graph empty is equivalent to finding an induced star partition. A follow-up study would involve determining which instances can be reduced to this special case, as well as broader analyses of how the problem behaves with different graph classes. We are particularly interested in studying its complexity for graph classes where the it does not admit a trivial answer.

Referências

Babenko, M. and Gusakov, A. (2013). New Exact and Approximation Algorithms for the Star Packing Problem in Undirected Graphs. *LIPIcs, Volume 9, STACS 2011*, 9:519–530.

Blum, N. (1990). A new approach to maximum matching in general graphs. In Paterson, M. S., editor, *Automata, Languages and Programming*, volume 443, pages 586–597. Springer-Verlag, Berlin/Heidelberg.

Divya, D. and Vijayakumar, S. (2024). On Star Partition of Split Graphs. In Kalyanasundaram, S. and Maheshwari, A., editors, *Algorithms and Discrete Applied Mathematics*, volume 14508, pages 209–223. Springer Nature Switzerland, Cham.

Edmonds, J. (1965). Paths, Trees, and Flowers. *Canadian Journal of Mathematics*, 17:449–467.

Gabow, H. N. and Tarjan, R. E. (1989). Faster Scaling Algorithms for Network Problems. *SIAM Journal on Computing*, 18(5):1013–1036.

Gallian, J. A. (2022). A Dynamic Survey of Graph Labeling. *The Electronic Journal of Combinatorics*, 1000:DS6: Dec 2.

Kelmans, A. K. (1997). Optimal packing of induced stars in a graph. *Discrete Mathematics*, 173(1-3):97–127.

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50.

Lovász, L. and Plummer, M. D. (2009). *Matching Theory*, volume 367. American Mathematical Soc.

Manber, U. (1998). *Introduction to algorithms: a creative approach*. Addison-Wesley, Reading, Mass., reprinted with corr., [nachdr.] edition.

Micali, S. and Vazirani, V. V. (1980). An O(v|v| c |E|) algoithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science (sfcs 1980)*, pages 17–27, Syracuse, NY, USA. IEEE.

Saito, A. and Watanbe, M. (1993). Partitioning Graphs into Induced Stars. *Ars Combinatoria*, Volume 036:3–6.

Shalu, M., Vijayakumar, S., Sandhya, T., and Mondal, J. (2022). Induced star partition of graphs. *Discrete Applied Mathematics*, 319:81–91.