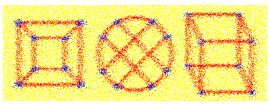
Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 19, no. 1, pp. 441–465 (2015)
DOI: 10.7155/jgaa.00371

Graph Orientations Optimizing the Number of Light or Heavy Vertices

Yuichi Asahiro[1] *Jesper Jansson*[2] *Eiji Miyano*[3] *Hirotaka Ono*[4]

[1]Department of Information Science, Kyushu Sangyo University, Matsukadai, Higashi-ku, Fukuoka 813-8503, Japan
[2]Laboratory of Mathematical Bioinformatics, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan
[3]Department of Systems Design and Informatics, Kyushu Institute of Technology, Iizuka, Fukuoka 820-8502, Japan
[4]Department of Economic Engineering, Kyushu University, Higashi-ku, Fukuoka 812-8581, Japan

Abstract

This paper introduces four graph orientation problems named Maximize W-Light, Minimize W-Light, Maximize W-Heavy, and Minimize W-Heavy, where W can be any fixed non-negative integer. In each problem, the input is an undirected, unweighted graph G and the objective is to assign a direction to every edge in G so that the number of vertices with outdegree at most W or at least W in the resulting directed graph is maximized or minimized. A number of results on the computational complexity and polynomial-time approximability of these problems for different values of W and various special classes of graphs are derived. In particular, it is shown that Maximize 0-Light and Minimize 1-Heavy are identical to Maximum Independent Set and Minimum Vertex Cover, respectively, so by allowing the value of W to vary, we obtain a new generalization of the two latter problems.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
May 2014	July 2015	August 2015	September 2015	September 2015

Published: October 2015

Article type: Regular paper

Communicated by: K. Kawarabayashi

A preliminary version of this article appeared in *Proceedings of the 2nd International Symposium on Combinatorial Optimization* (ISCO 2012), volume 7422 of *Lecture Notes in Computer Science*, pp. 332–343, Springer-Verlag Berlin Heidelberg, 2012.

This research was funded by KAKENHI grant numbers 23500020, 25330018, 26330017, and 26540005 and The Hakubi Project at Kyoto University.

E-mail addresses: asahiro@is.kyusan-u.ac.jp (Yuichi Asahiro) jj@kuicr.kyoto-u.ac.jp (Jesper Jansson) miyano@ces.kyutech.ac.jp (Eiji Miyano) hirotaka@econ.kyushu-u.ac.jp (Hirotaka Ono)

1 Introduction

Two well-studied computational problems in theoretical computer science are Maximum Independent Set and Minimum Vertex Cover. In these two problems, the input is an undirected graph $G = (V, E)$, and the objectives are to find a largest possible subset V' of V such that no two vertices in V' are adjacent in G (Maximum Independent Set) and a smallest possible subset V' of V such that every edge in E is incident to at least one vertex in V' (Minimum Vertex Cover). They were among the first problems ever to be shown to be NP-hard [1], and were used as a starting point for proving the NP-hardness of countless other problems during the 1970s [22] and onwards. In recent years, they have been central to the development of three important subfields of computational complexity: polynomial-time approximation algorithms, hardness of approximation, and parameterized complexity.

A relatively less researched area is the computational complexity of graph orientation problems. [2] By an *orientation* of an undirected graph G, we mean an assignment of a direction to each one of its edges. A *graph orientation problem* takes an undirected graph G as input and asks for an orientation of G that optimizes some well-defined criterion on the resulting directed graph, typically involving connectivity between vertices, the diameter, acyclicity, or constraints on the vertices' indegrees and/or outdegrees.

In this paper, we connect the concepts of Maximum Independent Set / Minimum Vertex Cover and graph orientation. We introduce four closely related graph orientation problems called Maximize W-Light, Minimize W-Light, Maximize W-Heavy, and Minimize W-Heavy, where W can be any fixed non-negative integer, and study their computational complexity and polynomial-time approximability for different values of W and different graph classes. Significantly, we demonstrate that Maximum Independent Set and Minimum Vertex Cover can be viewed as special cases of these graph orientation problems. Thus, by varying the parameter W, a new generalization of Maximum Independent Set and Minimum Vertex Cover is obtained.

1.1 Definitions

Let $G = (V, E)$ be an undirected, unweighted graph with a vertex set V and an edge set E. An *orientation* Λ *of* G is a function that maps each undirected edge $\{u, v\}$ in E to one of the two possible directed edges (u, v) and (v, u). Applying Λ to all edges in E transforms G into a directed graph, which we denote by $\Lambda(G)$. See Figure 1 for an example. For convenience, we write $\Lambda(E) = \bigcup_{e \in E}\{\Lambda(e)\}$ to refer to the set of directed edges in $\Lambda(G)$.

[1] In [30], Karp established the NP-hardness of several fundamental problems including Minimum Vertex Cover and Maximum Clique, which is computationally equivalent to Maximum Independent Set.

[2] Other aspects of graph orientations not directly related to computational complexity have been studied in graph theory and combinatorial optimization; see chapter 61 in [37] for a survey.

Figure 1: An example of an undirected graph G and an orientation Λ of G. Four vertices are 2-light and three vertices are 2-heavy in $\Lambda(G)$.

Next, for any vertex $u \in V$, define *the outdegree of u under Λ* as $d^+_\Lambda(u) = |\{v : (u, v) \in \Lambda(E)\}|$, i.e., the number of outgoing edges from u in the directed graph $\Lambda(G)$. For any non-negative integer W, a vertex $u \in V$ is said to be *W-light in $\Lambda(G)$* if $d^+_\Lambda(u) \leq W$, and *W-heavy in $\Lambda(G)$* if $d^+_\Lambda(u) \geq W$.

We now define four graph orientation problems that we call Maximize W-Light, Minimize W-Light, Maximize W-Heavy, and Minimize W-Heavy. In each problem, W is a fixed non-negative integer, the input is an undirected graph $G = (V, E)$ and the output is an orientation Λ of G such that:

- Maximize W-Light:
  the number of W-light vertices in $\Lambda(G)$ is maximized.
- Minimize W-Light:
  the number of W-light vertices in $\Lambda(G)$ is minimized.
- Maximize W-Heavy:
  the number of W-heavy vertices in $\Lambda(G)$ is maximized.
- Minimize W-Heavy:
  the number of W-heavy vertices in $\Lambda(G)$ is minimized.

Throughout the paper, we define $n = |V|$ and $m = |E|$. For any instance of Maximize W-Light or Minimize W-Light, let $OPT(G)$ denote the number of W-light vertices in an optimal solution, and for any instance of Maximize W-Heavy or Minimize W-Heavy, let $OPT(G)$ denote the number of W-heavy vertices in an optimal solution. Without loss of generality, the input graph G is assumed to be connected.

Let $\mathcal{A}$ be an algorithm that takes as input an undirected graph G and outputs an orientation of G. We say that $\mathcal{A}$ is a *σ-approximation algorithm* for Maximize W-Light (resp. Maximize W-Heavy), or that *$\mathcal{A}$'s approximation ratio is at most σ*, if $\mathcal{A}(G) \geq \frac{OPT(G)}{\sigma}$ holds for every input graph G, where $\mathcal{A}(G)$ is the number of W-light (resp. W-heavy) vertices in the solution returned by $\mathcal{A}$. Similarly, we say that $\mathcal{A}$ is a *σ-approximation algorithm* for Minimize W-Light (resp. Minimize W-Heavy), or that *$\mathcal{A}$'s approximation ratio is at*

most σ, if $\mathcal{A}(G) \leq \sigma \cdot OPT(G)$ holds for every input graph G, where $\mathcal{A}(G)$ is the number of W-light (resp. W-heavy) vertices in the solution returned by $\mathcal{A}$.

For any given graph $G = (V, E)$, any orientation Λ of G, and any fixed non-negative integer W, the vertex set V is partitioned into two disjoint subsets: the set of W-light vertices in $\Lambda(G)$ and the set of $(W+1)$-heavy vertices in $\Lambda(G)$. Therefore, if $\mathcal{A}$ is an algorithm that solves Maximize W-Light exactly then $\mathcal{A}$ solves Minimize $(W+1)$-Heavy exactly as well, and we say that for every fixed $W \geq 0$, Maximize W-Light and Minimize $(W+1)$-Heavy are *supplementary problems*. However, if $\mathcal{A}$ is a good approximation algorithm for Maximize W-Light, it does not automatically follow that $\mathcal{A}$ yields a good approximation algorithm for Minimize $(W+1)$-Heavy. The relationship between Minimize W-Light and Maximize $(W+1)$-Heavy is analogous. [3]

1.2 New results and organization of the paper

We derive some basic results on the computational complexity of the new graph orientation problems and investigate their relations to various computationally easy and hard problems. See the tables in Figure 2 for a summary. The paper is organized as follows:

- Section 2 considers Maximize W-Light and Minimize $(W+1)$-Heavy. We prove that the two problems coincide with Maximum Independent Set and Minimum Vertex Cover, respectively, when $W = 0$. Moreover, Section 2 shows that Maximize W-Light and Minimize $(W+1)$-Heavy are NP-hard for every fixed $W \geq 1$, and explains how to approximate the former for any $W \geq 1$ and the latter for $W = 1$ in polynomial time for any graph G and how to solve them exactly in linear time for any $W \geq 1$ when G is a tree.

- Section 3 is devoted to Minimize W-Light and Maximize $(W+1)$-Heavy. We show that when $W = 0$, the problems can be solved exactly in linear time for all classes of graphs. For unbounded W, they can be solved in quadratic time when restricted to outerplanar graphs and in linear time when restricted to trees. On the other hand, the problems become NP-hard for every fixed $W \geq 2$, even if restricted to planar graphs. In the general graph case, for any $W \geq 1$, we can approximate Minimize W-Light within a ratio of $(W+1)$ in polynomial time with a maximum flow-based technique, and Maximize $(W+1)$-Heavy within a ratio of $(W+2)$ in linear time with a greedy algorithm.

- Section 4 discusses open problems.

[3] Pairs of supplementary problems whose polynomial-time approximability properties differ greatly can be found in the literature. For example, Maximum Independent Set and Minimum Vertex Cover are supplementary problems, and it is known that Maximum Independent Set is NP-hard to approximate within a ratio of n^{ϵ} for any constant $0 \leq \epsilon < 1$ [39], while Minimum Vertex Cover can be approximated within a ratio of $2 - \Theta(\frac{1}{\sqrt{\log n}})$ in polynomial time [29].

Section 2	MAXIMIZE W-LIGHT	MINIMIZE $(W+1)$-HEAVY
$W = 0$	Identical to MAXIMUM INDEPENDENT SET (Theorem 1)	Identical to MINIMUM VERTEX COVER (Theorem 1)
$W \geq 0$	NP-hard (Theorem 3)	NP-hard (Theorem 3)
$W = 1$	$O(n^2)$-time $(\frac{n}{\log n})$-approx. (Corollary 3)	Polynomial-time 2-approx. (Theorem 5)
$W \geq 1$	Solvable in $O(n)$ time for trees (Theorem 2) and $\tilde{O}(n^2)$-time $(\frac{n}{\log n})$-approx. (Theorem 4)	Solvable in $O(n)$ time for trees (Theorem 2)
$W \geq 3$	Solvable in $O(n)$ time for planar graphs (Corollary 1)	Solvable in $O(n)$ time for planar graphs (Corollary 1)

Section 3	MINIMIZE W-LIGHT	MAXIMIZE $(W+1)$-HEAVY
$W = 0$	Solvable in $O(m)$ time (Theorem 6)	Solvable in $O(m)$ time (Theorem 6)
$W \geq 0$	Solvable in $O(n)$ time for trees (Theorem 7) and in $O(n^2)$ time for outerplanar graphs (Theorem 8)	Solvable in $O(n)$ time for trees (Theorem 7) and in $O(n^2)$ time for outerplanar graphs (Theorem 8)
$W \geq 1$	Polynomial-time $(W+1)$-approx. (Theorem 11)	$O(n^2)$-time 2-approx. for planar graphs (Theorem 8) and $O(m)$-time $(W+2)$-approx. for unrestricted graphs (Theorem 10)
$W \geq 2$	NP-hard even for planar graphs (Theorem 9)	NP-hard even for planar graphs (Theorem 9)

Figure 2: Summary of the results described in this paper. The omitted problem variant MINIMIZE 0-HEAVY is trivial because all vertices have outdegree ≥ 0 under every orientation of G, i.e., $OPT(G) = n$ always holds and any arbitrary orientation of the edges gives an optimal solution. In the same way, MAXIMIZE 0-HEAVY is trivial with $OPT(G) = n$.

1.3 Motivation

The main reason for introducing the above problems is that they provide a natural extension of the well-known MAXIMUM INDEPENDENT SET and MINIMUM VERTEX COVER problems. By studying the computational complexity of the new problems as W varies, one may gain novel insights into the structure of MAXIMUM INDEPENDENT SET and MINIMUM VERTEX COVER. For example,

the original Minimum Vertex Cover problem, i.e., Minimize $(W+1)$-Heavy with $W = 0$ according to Theorem 1 below, is NP-hard even if restricted to planar graphs [22]. By Corollary 1, it becomes solvable in polynomial time for planar graphs when $W \geq 3$. In contrast, Theorem 3 shows that increasing W does not decrease the computational complexity for unrestricted (non-planar) graphs.

An application of graph orientation problems in general is *load balancing* for parallel machine scheduling, where some "jobs" (corresponding to edges in G) have to be distributed among "machines" (corresponding to vertices in G) in a fair way [1, 3, 16], and each job must be handled by one of two specified machines. As an example, the problem of computing an orientation of G that minimizes the maximum outdegree was studied in [3, 4, 16, 38]; in the context of efficient data structures, if G is a sparse graph then an orientation of G that minimizes the maximum outdegree can be employed to support fast (in the worst case) vertex adjacency queries in G when using adjacency lists [9, 11]. As another example, a graph orientation of G that maximizes the minimum outdegree [1] solves a special case of the Santa Claus problem [7] in which Santa Claus has to distribute a set of gifts (corresponding to edges in G) among a set of children (corresponding to vertices in G), each gift is of value to exactly two children, and the objective is to make the least lucky child as happy as possible. The new problems Maximize W-Light and Minimize $(W+1)$-Heavy introduced in this paper can be regarded as load balancing problems where every machine initially has a capacity of W and we wish to find a job assignment that requires as few machines as possible to be upgraded to "supermachines" with unlimited capacity. As for Minimize W-Light and Maximize $(W+1)$-Heavy, these correspond to a version of the Santa Claus problem where each gift can be given to one of two specified children, a child is "happy" if he / she receives at least $W+1$ gifts, and one wants to maximize the number of happy kids.

1.4 Related work

We note that the algorithm in Section 4.2 of [4] computes an orientation Λ of G that minimizes $\max\{d^+_\Lambda(u) : u \in V\}$, taken over all possible orientations of G, in $O(m^{3/2} \cdot \log \Delta)$ time, where Δ is at most the maximum (unweighted) degree among all vertices in G. Using this algorithm, we can find an orientation of G under which *all* vertices are W-light, for any given W, when such an orientation exists. However, when such an orientation does not exist, the algorithm does not directly give a suitable solution for Maximize W-Light. Intuitively, in some instances of Maximize W-Light, it is better to "sacrifice" one vertex by giving it a high outdegree. As an example, let G be a star graph and $W = 0$. Orienting every edge towards the center vertex minimizes $\max\{d^+_\Lambda(u) : u \in V\}$, but gives a very poor solution for Maximize 0-Light. Nevertheless, we will make use of this algorithm in Section 2.4 below to approximate Maximize W-Light.

Similarly, Algorithm `Exact-1-MaxMinO` in Section 3 of [1] computes an orientation Λ of G maximizing $\min\{d^+_\Lambda(u) : u \in V\}$, taken over all orientations of G, in $O(m^{3/2} \cdot \log m \cdot (\log \Delta)^2)$ time. By running `Exact-1-MaxMinO`, it is

trivial to construct an orientation of G in which all vertices are W-heavy, for any given W, when one exists.

Every planar graph has an orientation in which all vertices are 3-light [11, 35], and Chrobak and Eppstein [11] gave two different linear-time algorithms for constructing such an orientation. Hence:

Corollary 1 *([11]) For any $W \geq 3$,* Maximize W-Light *and* Minimize $(W+1)$-Heavy *restricted to planar graphs can be solved in $O(n)$ time.*

Another pair of supplementary problems that generalize Maximum Independent Set and Minimum Vertex Cover when their parameter is allowed to vary are Maximum k-Dependent Set [14, 27] (sometimes called Maximum co-$(k+1)$-Plex [6, 33]) and Bounded-Degree-k Vertex Deletion [10, 18, 32, 34, 36], where k is a fixed non-negative integer. The former problem is to find a largest possible subset V' of the vertices in the input graph G such that each vertex in V' is adjacent to at most k vertices in V', and the latter is to find a smallest possible subset of the vertices in G whose removal transforms G into a graph of degree at most k. (Thus, Maximum Independent Set and Minimum Vertex Cover correspond to the special case $k = 0$.) Figure 3

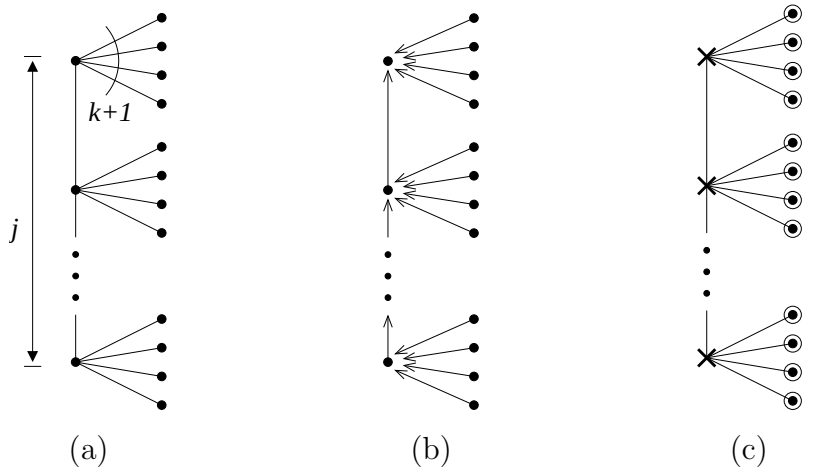

Figure 3: Let $k \geq 0$ and $j \geq 1$ be integers and let G be the tree in (a) obtained by attaching $k+1$ leaves to each node on a path of length j. Then $n = (k+2) \cdot j$. For every $W \geq 1$, the orientation in (b) has n vertices that are W-light and 0 vertices that are $(W+1)$-heavy, so this is an optimal solution to Maximize W-Light and Minimize $(W+1)$-Heavy. On the other hand, the optimal solutions to Maximum k-Dependent Set and Bounded-Degree-k Vertex Deletion indicated by circles and crosses, respectively, in (c) have cardinalities $(k+1) \cdot j$ and j.

shows that these generalizations are different from MAXIMIZE W-LIGHT and MINIMIZE $(W+1)$-HEAVY.

A previously studied optimization problem related to *partial* graph orientation under degree constraints is: Given an undirected graph with specified upper bounds on the indegree and the outdegree of each vertex, orient as many edges as possible while complying with the degree constraints (i.e., some edges might be left unoriented in the solution). Gabow [20] showed that this problem is MAXSNP-hard and 4/3-approximable in polynomial time.

2 MAXIMIZE W-LIGHT & MINIMIZE $(W+1)$-HEAVY

This section investigates the supplementary problems MAXIMIZE W-LIGHT and MINIMIZE $(W+1)$-HEAVY for different values of W.

2.1 $W = 0$

We first prove the following lemma:

Lemma 1 *Let $G = (V, E)$ be an undirected graph. For any orientation Λ of G, the set of 0-light vertices in $\Lambda(G)$ forms an independent set in G. Conversely, given any independent set I in G, there exists an orientation of G in which the vertices from I are 0-light.*

Proof: $\Longrightarrow$) For any pair u, v of 0-light vertices in $\Lambda(G)$, no edges in $\Lambda(G)$ are oriented away from u or v by the definition of 0-light, so G cannot contain the edge $\{u, v\}$. Thus, the set of 0-light vertices forms an independent set in G.

$\Longleftarrow$) Define an orientation Λ of G as follows. First, for each $u \in I$, orient all edges involving u towards u. Next, orient all remaining edges arbitrarily. Obviously, every vertex from I will be 0-light in $\Lambda(G)$. □

As mentioned in Section 1.1, for any orientation Λ of G, the set of 0-light vertices in $\Lambda(G)$ and the set of 1-heavy vertices in $\Lambda(G)$ form a partition of V. Also, any subset $V' \subseteq V$ is an independent set in G if and only if $V \setminus V'$ is a vertex cover of G. Together with Lemma 1, this yields:

Lemma 2 *Let $G = (V, E)$ be an undirected graph. For any orientation Λ of G, the set of 1-heavy vertices in $\Lambda(G)$ forms a vertex cover of G. Conversely, given any vertex cover C of G, there exists an orientation of G in which the vertices from C are 1-heavy.*

Hence:

Theorem 1 MAXIMIZE 0-LIGHT *and* MAXIMUM INDEPENDENT SET *are identical, and* MINIMIZE 1-HEAVY *and* MINIMUM VERTEX COVER *are identical.*

Consequently, the known hardness results for Maximum Independent Set [39] and Minimum Vertex Cover [15] immediately carry over to Maximize 0-Light and Minimize 1-Heavy. On the positive side, we can apply existing approximation algorithms for Maximum Independent Set [17] and Minimum Vertex Cover [29]. Furthermore, Maximum Independent Set and Minimum Vertex Cover can be solved in polynomial time for some classes of graphs such as bipartite graphs [24], and even in linear time for certain useful special classes of graphs such as chordal graphs[4] [23]. In summary, we have:

Corollary 2
- *([39])* Maximize 0-Light *cannot be approximated within a ratio of* n^ϵ *for any constant* $0 \leq \epsilon < 1$ *in polynomial time, unless* $P = NP$.
- *([17])* Maximize 0-Light *can be approximated within a ratio of* $O(n(\log\log n)^2/(\log n)^3)$ *in polynomial time.*
- *([15])* Minimize 1-Heavy *cannot be approximated within a ratio of* 1.3606 *in polynomial time, unless* $P = NP$.
- *([29])* Minimize 1-Heavy *can be approximated within a ratio of* $2 - \Theta(\frac{1}{\sqrt{\log n}})$ *in polynomial time.*
- *([24])* Maximize 0-Light *and* Minimize 1-Heavy *restricted to bipartite graphs can be solved in polynomial time.*
- *([23])* Maximize 0-Light *and* Minimize 1-Heavy *restricted to chordal graphs can be solved in linear time.*

2.2 $W \geq 1$, restriction to trees

When G is a tree, optimal solutions to the problems can be computed easily by applying an algorithm named `Up-To-Roots` in [3] that works as follows:

> Select any node r in G, root G in r, and orient every edge towards r.

Clearly, `Up-To-Roots` produces an orientation with exactly $n-1$ vertices having outdegree 1 and one vertex having outdegree 0, which means that for any $W \geq 1$, trivially, all n vertices are W-light and none are $(W+1)$-heavy. This gives:

Theorem 2 *For any* $W \geq 1$, Maximize W-Light *and* Minimize $(W+1)$-Heavy *restricted to trees can be solved in* $O(n)$ *time.*

[4]The class of chordal graphs includes many famous classes such as interval graphs, split graphs, threshold graphs, and trees [8].

2.3 $W \geq 1$, NP-hardness

By Corollary 2 above, MAXIMIZE W-LIGHT and MINIMIZE $(W+1)$-HEAVY are NP-hard when $W = 0$. We now show that the problems are also NP-hard for other values of W.

Theorem 3 *For every fixed $W \geq 0$,* MAXIMIZE W-LIGHT *and* MINIMIZE $(W+1)$-HEAVY *are NP-hard.*

Proof: We give a polynomial-time reduction from MAXIMIZE W-LIGHT to MAXIMIZE $(W+1)$-LIGHT for any $W \geq 0$. The theorem then follows from the NP-hardness of MAXIMIZE 0-LIGHT, induction, and MAXIMIZE W-LIGHT and MINIMIZE $(W+1)$-HEAVY being supplementary. Let $G = (V, E)$ be a given instance of MAXIMIZE W-LIGHT. Construct an instance $G' = (V', E')$ of MAXIMIZE $(W+1)$-LIGHT by taking $|V|$ copies of the complete graph K_{2W+3} with $2W+3$ vertices and attaching one such copy K^i to each $v_i \in V$ by adding an edge between v_i and any one vertex w_i in K^i. We claim that G has an orientation in which at least k vertices are W-light if and only if G' has an orientation in which at least $k + (2W+3) \cdot |V|$ vertices are $(W+1)$-light.

To prove the claim, first note that if G has an orientation with at least k vertices that are W-light, it is straightforward to extend it to an orientation of G' that makes at least $k + (2W+3) \cdot |V|$ vertices $(W+1)$-light by orienting, for every $v_i \in V$: (i) the edge $\{v_i, w_i\}$ as (v_i, w_i); and (ii) the edges inside K^i so that all of its $2W+3$ vertices get outdegree $(W+1)$ (for example, denote the vertices of K^i by $\{u_0, u_1, \ldots, u_{2W+2}\}$ and orient each edge $\{u_j, u_k\}$ as (u_j, u_k) if $k - j \pmod{2W+3} \in \{1, 2, \ldots, W+1\}$ and as (u_k, u_j) otherwise). Next, suppose that G' has an orientation in which at least $k + (2W+3) \cdot |V|$ vertices are $(W+1)$-light. For every $v_i \in V$, if some vertex belonging to K^i is not $(W+1)$-light then orienting $\{v_i, w_i\}$ and the edges of K^i as in (i) and (ii) above yields an orientation Λ' of G' in which all vertices in $V' \setminus V$ are $(W+1)$-light while not decreasing the total number of $(W+1)$-light vertices. At least k vertices in V are $(W+1)$-light in Λ' and for each such v_i, the edge $\{v_i, w_i\}$ is oriented as (v_i, w_i). Thus, taking the restriction of Λ' to the original G gives an orientation of G in which at least k vertices are W-light. □

2.4 $W \geq 1$, approximation algorithms

To approximate MAXIMIZE W-LIGHT for any fixed $W \geq 1$ as well as MINIMIZE 2-HEAVY in polynomial time, we will apply some known results from the literature.

We first consider MAXIMIZE W-LIGHT. To approximate it, we use the simple partitioning technique from Proposition 2.2 in [26] to get an $(\frac{n}{\log n})$-approximation. It will be referred to as Algorithm `Partition_into_subsets` below. For clarity, the algorithm is shown explicitly in Figure 4.

Theorem 4 *For any $W \geq 1$, Algorithm* `Partition_into_subsets` *is an $(\frac{n}{\log n})$-approximation algorithm for* MAXIMIZE W-LIGHT *and runs in $\tilde{O}(n^2)$ time.*

1. Arbitrarily partition V into $\lfloor n/\log n \rfloor$ sets $V_1, V_2, ..., V_{\lfloor n/\log n \rfloor}$, each of size at most $\lceil \log n \rceil + 1$.
2. For every subset V_i' of every set V_i, check if the subgraph of G induced by V_i' has an orientation in which all vertices are W-light. Let Z be any such subset of maximum cardinality, and let Λ be the corresponding orientation of the subgraph of G induced by Z.
3. Extend Λ to all of G by orienting all edges with exactly one endpoint in Z towards Z and orient all remaining edges of G arbitrarily.

Figure 4: Algorithm `Partition_into_subsets`.

Proof: According to Steps 2 and 3, all vertices belonging to Z are guaranteed to be W-light in the resulting orientation of G.

To analyze the approximation ratio, we use the same argument as in the analysis of Proposition 2.2 in [26]. Let Λ^* be any optimal orientation of G and let Z^* be the set of W-light vertices in Λ^*. By the pigeonhole principle, at least one of the subsets $V_1, V_2, ..., V_{\lfloor n/\log n \rfloor}$ contains a fraction of $\frac{1}{\lceil n/\log n \rceil}$ or more of the elements in Z^*. For any $V_i' \subseteq V$, every vertex in V_i' that is W-light in Λ^* will still be W-light in the subgraph of G induced by V_i' because its outdegree will not increase. Thus, the discovered Z satisfies $|Z| \geq \frac{|Z^*|}{\lceil n/\log n \rceil} \geq \frac{|Z^*|}{n/\log n}$.

The running time is $O(\frac{n}{\log n} \cdot n \cdot \log^3 n \log\log n) = O(n^2 \log^2 n \log\log n) = \tilde{O}(n^2)$ because every V_i has at most $2^{\log n + 2} = O(n)$ subsets, and each such V_i' can be tested in Step 2 to see if the subgraph of G induced by V_i' admits an orientation that makes all vertices W-light in $O(\log^3 n \log\log n)$ time by the algorithm in Section 4.2 of [4], which computes an orientation Λ' of a given graph $G' = (V', E')$ that minimizes $\max\{d^+_{\Lambda'}(u) : u \in V'\}$ taken over all possible orientations of G' in $O(|E'|^{3/2} \cdot \log \Delta')$ time, where Δ' is at most the maximum (unweighted) degree among all vertices in V' (here, $|V'| = |V_i'| = O(\log n)$). □

Next, we focus on the case $W = 1$. Recall the following definitions from the literature (see, e.g., [21]): A *pseudotree* is a connected, undirected graph containing exactly one cycle, and a *pseudoforest* is a union of vertex-disjoint trees and pseudotrees. Thus, every connected component of a pseudoforest has at most one cycle. For any subset $V' \subseteq V$ of the vertices in a graph $G = (V, E)$, V' *induces* the subgraph of G consisting of V' and all edges in E whose two endpoints belong to V'. An induced subgraph of G that is a pseudoforest and has the largest possible number of vertices is a *maximum induced pseudoforest in* G. The next lemma is analogous to Lemma 1:

Lemma 3 *Let $G = (V, E)$ be an undirected graph. For any orientation Λ of G, the subgraph of G induced by the set of 1-light vertices in $\Lambda(G)$ is a pseudoforest. Conversely, for any subset $V' \subseteq V$, if V' induces a pseudoforest in G then there exists an orientation of G in which the vertices from V' are 1-light.*

Proof: $\Longrightarrow$) Let G' be the subgraph of G induced by the set of 1-light vertices

in $\Lambda(G)$. In every connected component X in G', the number of edges is less than or equal to the number of vertices by the definition of a 1-light vertex, so if X is not a tree then X contains precisely one cycle. Therefore, each connected component in G' is either a tree or a pseudotree, and it follows that G' is a pseudoforest.

$\Longleftarrow$) Define an orientation Λ of G as follows. For each connected component X in the induced pseudoforest, if X is a tree then apply `Up-To-Roots` from Section 2.2 to X; otherwise, let C be the cycle in X, orient all edges of X not belonging to C towards C, and traverse C in either direction while orienting all its edges forward. Finally, orient every edge in G with exactly one endpoint in V' towards V' and orient all remaining edges of G arbitrarily. This way, every vertex in V' will be 1-light in $\Lambda(G)$. □

Lemma 3 implies that the running time of Algorithm `Partition_into_subsets` can be improved slightly for the special case $W = 1$:

Corollary 3 MAXIMIZE 1-LIGHT *can be approximated within a ratio of* $(\frac{n}{\log n})$ *in* $O(n^2)$ *time.*

Proof: Proceed as in the proof of Theorem 4, but implement Step 2 as follows. To check if the subgraph of G induced by V_i' has an orientation in which all vertices are W-light, check if V_i' induces a pseudoforest in G in $O(|V_i'|) = O(\log n)$ time by depth-first search instead of using the algorithm from [4]. The total running time will be $O(\frac{n}{\log n} \cdot n \cdot \log n) = O(n^2)$. □

More importantly, it also follows from Lemma 3, together with the fact that any given orientation of G partitions V into 1-light and 2-heavy vertices, that MINIMIZE 2-HEAVY is identical to the problem of finding a smallest cardinality subset of V whose removal leaves a pseudoforest. This is a special case of the more general "node-deletion problem for a graph property π" in [19], which asks for a minimum set of vertices whose deletion leaves a subgraph satisfying π; more precisely, it is the special case where π is the property "every connected component of a graph contains at most one cycle". By point (4) of Theorem 9 in [19], when restricted to this particular property, the output of the polynomial-time approximation algorithm in [19] is at most twice the cardinality of a minimum solution. Therefore:

Theorem 5 MINIMIZE 2-HEAVY *has a polynomial-time* 2*-approximation algorithm.*

3 MINIMIZE W-LIGHT & MAXIMIZE $(W+1)$-HEAVY

We now consider the pair of supplementary problems MINIMIZE W-LIGHT and MAXIMIZE $(W+1)$-HEAVY.

3.1 $W = 0$

For $W = 0$, the following theorem holds.

Theorem 6 MINIMIZE 0-LIGHT *and* MAXIMIZE 1-HEAVY *can be solved in* $O(m)$ *time.*

Proof: Let $G = (V, E)$ be the input undirected graph. There are two cases:

1. If G is a tree then G has n vertices and $n-1$ edges, so at most $n-1$ vertices can be 1-heavy in any orientation of G. Run the algorithm `Up-To-Roots` mentioned in Section 2.2 to obtain an orientation of G with $n-1$ 1-heavy vertices, which is optimal, in $O(n) = O(m)$ time.

2. If G is not a tree then first find a cycle C in G by depth-first search in $O(m)$ time. Then, select an arbitrary edge e on C and orient it away from u, where u is either one of its endpoints. Compute any spanning tree T of $G \setminus \{e\}$ in $O(m)$ time, root T at the vertex u, and orient every edge of T towards the root. This ensures that every vertex of G gets outdegree at least 1, i.e., no vertex is 0-light and all vertices are 1-heavy in the resulting orientation of G. Finally, orient any remaining edges arbitrarily.

□

3.2 Polynomial-time solutions for special graphs

In this subsection, we allow W to be unbounded but restrict the input to certain graph classes.

We first present a linear-time solution for the case where the input G is a tree. The algorithm is listed in Figure 5. It is named `Extended_Up-To-Roots` because when $W = 0$, it is the same as the algorithm `Up-To-Roots` of [3] described in Section 2.2 if we always select a leaf as the node r.

1. Root the tree G in any leaf r, and for every $u \in V$, initialize $s(u) = 0$.
2. For every non-root node u in bottom-up order, do:
2.1 Let p be the parent of u.
2.2 If $s(u) = W$ then orient the edge $\{u, p\}$ from u to p.
2.3 Else, orient the edge $\{u, p\}$ from p to u and let $s(p) = s(p) + 1$.

Figure 5: Algorithm `Extended_Up-To-Roots` for solving MINIMIZE W-LIGHT and MAXIMIZE $(W + 1)$-HEAVY restricted to trees.

Theorem 7 *For any* $W \geq 0$*, Algorithm* `Extended_Up-To-Roots` *solves* MINIMIZE W-LIGHT *and* MAXIMIZE $(W + 1)$-HEAVY *restricted to trees in* $O(n)$ *time.*

Proof: For any non-root node $u \in V$, denote the subtree of G rooted at u by $G[u]$. Furthermore, denote the subtree of G consisting of the parent p of u, the undirected edge $\{u, p\}$, and $G[u]$ by $G^{\wedge}[u]$. We shall prove that after u has been treated in Step 2 of the algorithm, the orientation obtained so far gives an optimal solution for MAXIMIZE $(W+1)$-HEAVY for $G^{\wedge}[u]$. By the bottom-up ordering of the nodes, we may assume inductively that this property holds for all children of u (if any). Two cases are possible:

- u is a leaf: Then $OPT(G^{\wedge}[u])$ is either 1 (if $W = 0$) or 0 (if $W \geq 1$). When $W = 0$, Step 2.2 makes u one of the $(W+1)$-heavy vertices in the resulting orientation.

- u is an internal node: Let Λ be the orientation of $G^{\wedge}[u]$ constructed in Steps 2.2 and 2.3, and let $C(u)$ be the set of children of u. By the induction hypothesis, it holds that for each $u_i \in C(u)$, Λ restricted to $G^{\wedge}[u_i]$ gives an optimal orientation for $G^{\wedge}[u_i]$. Thus, the number of $(W+1)$-heavy vertices in an optimal solution for $G^{\wedge}[u]$ equals either $1 + \sum_{u_i \in C(u)} OPT(G^{\wedge}[u_i])$ (if u can be made $(W+1)$-heavy) or $\sum_{u_i \in C(u)} OPT(G^{\wedge}[u_i])$ (otherwise). (It is impossible for $G^{\wedge}[u]$ to have $2 + \sum_{u_i \in C(u)} OPT(G^{\wedge}[u_i])$ or more $(W+1)$-heavy vertices since it is a tree.) If $s(u) = W$ in Step 2.2, there are exactly W edges oriented from u to the children of u, and we make u a $(W+1)$-heavy vertex by orienting the edge $\{u, p\}$ towards p, so the obtained orientation of $G^{\wedge}[u]$ has $1 + \sum_{u_i \in C(u)} OPT(G^{\wedge}[u_i])$ vertices that are $(W+1)$-heavy. On the other hand, if $s(u) < W$ then u can not become $(W+1)$-heavy and if $s(u) > W$ then u is already $(W+1)$-heavy; in both of these cases, $G^{\wedge}[u]$ is already optimally oriented, and we orient $\{u, p\}$ away from p and increment $s(p)$ in Step 2.3.

The running time is $O(n+m) = O(n)$ because every node and every edge is considered once in the loop in Step 2 and because G is a tree. □

Next, we turn to the case where G is an outerplanar or planar graph. We reduce MAXIMIZE W-HEAVY to a problem named MAXIMUM Q-EDGE PACKING, studied in, e.g., [28]. The latter is defined as follows. Let Q be an undirected graph. The MAXIMUM Q-EDGE PACKING problem takes as input an undirected graph H (called a "host graph") and asks for the maximum number of edge-disjoint isomorphic copies of Q in H. Now, given an instance $G = (V, E)$ of MAXIMIZE W-HEAVY with n vertices, construct an instance of MAXIMUM $K_{1,(W+n)}$-EDGE PACKING, where $K_{1,(W+n)}$ denotes the star graph consisting of one center vertex with $W+n$ neighboring leaves. To do this, let H be a copy of the graph G, and for each $v \in V$, create n new vertices and attach them to v in H. See Figure 6 for an illustration. Thus, H contains $n + n^2$ vertices and $O(n^2)$ edges. Then, we have the following:

Lemma 4 *For any positive integer x, the graph G has an orientation Λ such that $\Lambda(G)$ contains at least x W-heavy vertices if and only if the graph H contains at least x edge-disjoint copies of $K_{1,(W+n)}$.*

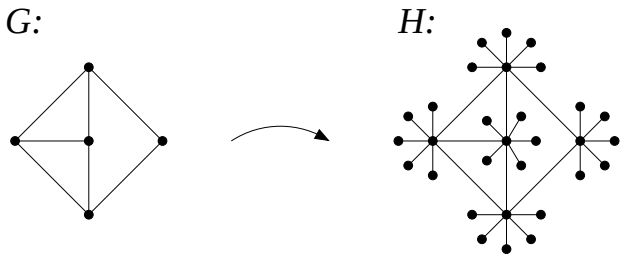

Figure 6: In the reduction from MAXIMIZE W-HEAVY to MAXIMUM $K_{1,(W+n)}$-EDGE PACKING, n new vertices are attached to each vertex in a copy of G to obtain the host graph H.

By transforming G to H as explained above and then applying the algorithms from [28] for MAXIMUM Q-EDGE PACKING, where Q is a star graph, we obtain:

Theorem 8
- *For any $W \geq 0$,* MINIMIZE W-LIGHT *and* MAXIMIZE $(W+1)$-HEAVY *restricted to outerplanar graphs can be solved in $O(n^2)$ time.*
- *For any $W \geq 0$,* MAXIMIZE $(W+1)$-HEAVY *restricted to planar graphs has an $O(n^2)$-time 2-approximation algorithm.*

Proof: If the given instance G of MINIMIZE W-LIGHT / MAXIMIZE $(W+1)$-HEAVY is an outerplanar graph then the constructed graph H is also outerplanar. According to Theorem 4.2 in [28], MAXIMUM Q-EDGE PACKING restricted to outerplanar graphs can be solved in $O(|V_H|)$ time by dynamic programming, where $|V_H|$ is the number of vertices in H, when Q is any star graph with at least 3 leaves. By Lemma 4 above, setting $Q = K_{1,(W+1+n)}$ and running the algorithm in Theorem 4.2 in [28] on H solves MINIMIZE W-LIGHT / MAXIMIZE $(W+1)$-HEAVY in $O(n^2)$ time.

In the more general case where G is a planar graph, H becomes planar, and we apply Theorem 5.3 in [28] instead, which says that MAXIMUM Q-EDGE PACKING restricted to planar graphs admits an $O(|E_H|)$-time 2-approximation algorithm when Q is a star graph with at least 3 leaves and where $|E_H|$ is the number of edges in H. □

3.3 $W \geq 2$, NP-hardness

Theorem 3.1 in [28] proves that for every fixed $W \geq 2$, MAXIMUM $K_{1,(W+1)}$-EDGE PACKING is NP-hard, even if restricted to planar graphs. (Recall that $K_{1,(W+1)}$ denotes the star graph with one center vertex and $W+1$ leaves.) The reduction is from PLANAR 3-SAT. We observe that in the reduction, every vertex in the constructed graph H has degree strictly less than $2(W+1)$. Therefore, any two copies of $K_{1,W+1}$ in H must use different center vertices, and it follows that any set of x edge-disjoint copies of the star graph $K_{1,(W+1)}$

in H induces an orientation of H in which x vertices are $(W+1)$-heavy, and vice versa. Hence, optimal solutions to the two problems MAXIMIZE $(W+1)$-HEAVY and MAXIMUM $K_{1,(W+1)}$-EDGE PACKING for the constructed graph H are equivalent. We obtain:

Theorem 9 *For every fixed $W \geq 2$,* MINIMIZE W-LIGHT *and* MAXIMIZE $(W+1)$-HEAVY *are NP-hard, even if restricted to planar graphs.*

3.4 $W \geq 1$, a greedy approximation algorithm for MAXIMIZE $(W+1)$-HEAVY

Figure 7 presents an approximation algorithm named `Greedy_Graph_Orientation` for MAXIMIZE $(W+1)$-HEAVY for general graphs. It runs in linear time.

1. Repeat until all vertices have been considered exactly once:
   1.1 Select any previously unconsidered vertex u.
   1.2 If u has at least $W+1$ incident unoriented edges then orient any $W+1$ of them away from u.
2. Orient any remaining unoriented edges arbitrarily.

Figure 7: Algorithm `Greedy_Graph_Orientation`.

Theorem 10 *For any $W \geq 1$, Algorithm* `Greedy_Graph_Orientation` *is a $(W+2)$-approximation algorithm for* MAXIMIZE $(W+1)$-HEAVY *and runs in $O(m)$ time.*

Proof: Let Λ be the partial orientation on E defined after Step 1 of the algorithm, where "partial" means that we allow some edges in E to remain unoriented. Let Λ^* be any optimal solution. Denote the set of $(W+1)$-heavy vertices in $\Lambda(G)$ by S and the set of $(W+1)$-heavy vertices in $\Lambda^*(G)$ by S^*. For any u in S, u takes an edge from at most $(W+1)$ vertices that belong to S^* and therefore prevents at most $(W+1)$ of the vertices in S^* from being $(W+1)$-heavy in $\Lambda(G)$. Denote the set of these vertices by S^*_u. Then, S^* can be partitioned into $\bigcup_{u\in S} S^*_u$ and $S \cap S^*$ (this is because if there is a vertex in $S^* \setminus (\bigcup_{u\in S} S^*_u \cup (S \cap S^*))$, it must be selected in Step 1.2 and thus be included in $S \cap S^*$, which would give a contradiction). Hence, the approximation ratio is:

$$\frac{|S^*|}{|S|} = \frac{|\bigcup_{u\in S} S^*_u| + |S \cap S^*|}{|S|} \leq \frac{\sum_{u\in S}|S^*_u| + |S|}{|S|} \leq \frac{\sum_{u\in S}(W+1) + |S|}{|S|} = W+2$$

Step 1 is performed exactly n times, and each edge is considered at most two times in total, so the algorithm can be implemented to run in $O(n+m) = O(m)$ time. □

The approximation ratio $(W+2)$ in Theorem 10 is tight. See Figure 8.

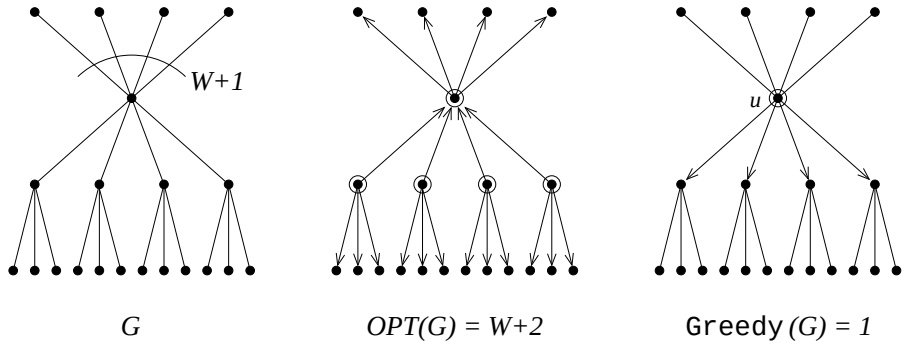

Figure 8: A bad example for Algorithm `Greedy_Graph_Orientation`. (In the figure, $W = 3$.) Here $OPT(G) = W + 2$, but if the algorithm first selects vertex u and orients $(W + 1)$ edges away from u as shown on the right, then only u can be $(W + 1)$-heavy in any resulting orientation.

We remark that Theorem 5.1 in [28] describes an approximation algorithm for Maximum Q-Edge Packing which is based on a similar idea. However, we cannot apply it here directly because it assumes that Q is fixed; it identifies all copies of Q in H, so its running time depends exponentially on the size of Q.

3.5 $W \geq 1$, an approximation algorithm for Minimize W-Light

Finally, we give a polynomial-time approximation algorithm for Minimize W-Light for any fixed $W \geq 1$. It is based on computing maximum flows in a family of flow networks $\{\mathcal{N}_G(0), \mathcal{N}_G(1), \ldots, \mathcal{N}_G(n)\}$ with positive edge capacities.

Let $G = (V, E)$ be any input undirected graph to Minimize W-Light. Define a directed graph $\mathcal{N}_G = (V_G, E_G)$ with vertex set V_G and edge set E_G as shown in Figure 9 by setting:

$$\begin{aligned} V_G &= V \cup E \cup \{r, s, t, z\} \cup \{x_1, x_2, \ldots, x_W, x_{W+1}\}, \\ E_G &= \big\{(s, v) \mid v \in V\big\} \cup \big\{(v_i, e), (v_j, e) \mid e = \{v_i, v_j\} \in E\big\} \cup \\ &\quad \big\{(e, z) \mid e \in E\big\} \cup \big\{(v, x_i) \mid v \in V,\ 1 \leq i \leq W + 1\big\} \cup \\ &\quad \big\{(x_i, r) \mid 1 \leq i \leq W + 1\big\} \cup \big\{(r, z), (z, t)\big\} \end{aligned}$$

(Each vertex in $\mathcal{N}_G$ that corresponds to a vertex v in G has outdegree $\deg(v) + W + 1$, and each vertex in $\mathcal{N}_G$ that corresponds to an edge in G has indegree 2.)

Next, for any integer $q \in \{0, 1, \ldots, n\}$, let $\mathcal{N}_G(q) = (V_G, E_G, cap_q)$ be the

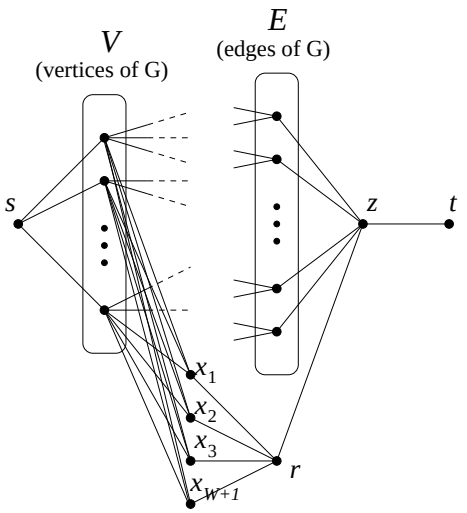

Figure 9: The directed graph $\mathcal{N}_G$. All edges are directed from left to right.

flow network obtained by augmenting $\mathcal{N}_G$ with edge capacities cap_q, where:

$$cap_q(a) = \begin{cases} W+1, & \text{if } a = (s, v) \text{ with } v \in V; \\ 1, & \text{if } a = (v, e) \text{ with } v \in V,\ e \in E; \\ 1, & \text{if } a = (v, x_i) \text{ with } v \in V,\ 1 \leq i \leq W+1; \\ 1, & \text{if } a = (e, z) \text{ with } e \in E; \\ q, & \text{if } a = (x_i, r) \text{ with } 1 \leq i \leq W+1; \\ (W+1) \cdot n, & \text{if } a = (r, z) \text{ or } a = (z, t). \end{cases}$$

Lemma 5 *For any $q \in \{0, 1, \ldots, n\}$, if the maximum directed flow from vertex s to vertex t in $\mathcal{N}_G(q)$ equals $(W+1) \cdot n$ then there exists an orientation of G in which the number of W-light vertices is at most $(W+1) \cdot q$.*

Proof: Fix q and let $\mathcal{F}$ be any maximum directed flow from s to t in $\mathcal{N}_G(q)$ with integer values. Suppose $\mathcal{F}$ has value $(W+1) \cdot n$. Construct an orientation Λ of G as follows. Every vertex in $\mathcal{N}_G$ that corresponds to an edge $e = \{u, v\}$ in E can receive at most one unit of flow in $\mathcal{F}$, and this will arrive at e either along (u, e), in which case we orient e in Λ as (u, v), or along (v, e), in which case we orient e in Λ as (v, u). Next, orient all remaining unoriented edges of G arbitrarily. Observe that for any $v \in V$, if the corresponding vertex in $\mathcal{N}_G$ does not send any of its $(W+1)$ units of flow to the vertices $\{x_1, x_2, \ldots, x_W, x_{W+1}\}$ then v is $(W+1)$-heavy in $\Lambda(G)$.

By the construction of $\mathcal{N}_G(q)$, at most $(W+1) \cdot q$ units of flow can pass through vertex r. Each of the n vertices in $\mathcal{N}_G$ that corresponds to a vertex in V receives precisely $W+1$ units of flow from s in $\mathcal{F}$, and at most $(W+1) \cdot q$ of them send at least one unit of flow to $\{x_1, x_2, \ldots, x_W, x_{W+1}\}$ and may therefore

not be $(W+1)$-heavy in $\Lambda(G)$. In other words, at most $(W+1)\cdot q$ vertices are W-light in $\Lambda(G)$. □

Lemma 6 *For any $q \in \{0,1,\ldots,n\}$, if the maximum directed flow from vertex s to vertex t in $\mathcal{N}_G(q)$ is strictly less than $(W+1)\cdot n$ then the number of W-light vertices in every orientation of G is strictly larger than q.*

Proof: We prove the contrapositive. Suppose there exists an orientation Λ of G with at most q W-light vertices. Construct a flow from s to t in $\mathcal{N}_G(q)$ with value $(W+1)\cdot n$ as follows.

First, use all of the capacity of the edges of the form (s,v), (e,z), (z,t), where $v \in V$ and $e \in E$. Then, for each $e = \{u,v\} \in E$, do the following: if $\Lambda(e) = (u,v)$ and the amount of already determined outgoing flow from u is less than $W+1$ then send one unit of flow along (u,e), and if $\Lambda(e) = (v,u)$ and the amount of already determined outgoing flow from v is less than $W+1$ then send one unit of flow along (v,e). Since at most q vertices are W-light, the total flow between vertices in $\mathcal{N}_G$ that correspond to vertices in V and vertices in $\mathcal{N}_G$ that correspond to edges in E is at least $(W+1)\cdot n - (W+1)\cdot q$. Next, for each W-light vertex v, distribute whatever remains of its at most $(W+1)$ units of flow among $\{x_1, x_2, \ldots, x_W, x_{W+1}\}$ arbitrarily by using capacity-1 edges of the form (v, x_i). Finally, let the flow along each edge (x_i, r) be the sum of all incoming flows to x_i and let the flow along (r,z) be the sum of all incoming flows to r. Thus, in total, the flow from s to t is $(W+1)\cdot n$. □

We now describe the algorithm. For any $q \in \{0,1,\ldots,n\}$, let $F(q)$ be an integral maximum directed flow from vertex s to vertex t in $\mathcal{N}_G(q)$, as computed by the algorithm of Goldberg and Rao [25]. [5]

1. Construct $\mathcal{N}_G$.
2. Let $q = 0$.
3. Repeat until the value of the flow $F(q)$ equals $(W+1)\cdot n$:
   $q = q + 1$.
4. For every $e \in E$, if an edge of the form (v_i, e) in $\mathcal{N}_G(q)$ has one unit of flow in $F(q)$ then orient e away from v_i in G. Orient all remaining unoriented edges of G arbitrarily.

Let $\mathcal{A}(G)$ be the number of W-light vertices in the orientation constructed by the algorithm. During the execution of the algorithm, at some point p, the following situation occurs:

(a) $q = p-1$ implies that the maximum flow in $\mathcal{N}_G(q) < (W+1)\cdot n$, and

(b) $q = p$ implies that the maximum flow in $\mathcal{N}_G(q) = (W+1)\cdot n$.

[5] Since all edge capacities are integers, we may assume by the integrality theorem (see, e.g., [12]) that the flow along each edge in $F(q)$ found by the algorithm in [25] is an integer.

By Lemma 6, (a) means that $OPT(G) > p - 1$, i.e., $OPT(G) \geq p$. By Lemma 5, (b) gives $\mathcal{A}(G) \leq (W+1) \cdot p$. It follows that the approximation ratio is at most $\frac{(W+1)\cdot p}{p} = W + 1$. We have just shown:

Theorem 11 *For any $W \geq 1$,* Minimize W-Light *can be approximated within a ratio of $(W + 1)$ in polynomial time.*

Remark: For greater efficiency, do a binary search on q in Step 3 instead of checking all candidate values of q incrementally. This gives an $O(n^3 \log^3 n)$-time algorithm.

4 Concluding remarks

We have introduced four new graph orientation problems and shown how they generalize the fundamental problems Maximum Independent Set and Minimum Vertex Cover in a natural way. For example, one interpretation of Minimize $(W+1)$-Heavy as a relaxed variant of Minimum Vertex Cover is that every vertex in the input graph is allowed to cover W or less of its incident edges "for free", without having to be placed in the output vertex cover.

The two tables in Figure 2 in Section 1.2 expose several open problems:

- What are the best possible polynomial-time approximation ratios for Maximize W-Light and Minimize $(W + 1)$-Heavy with $W \geq 1$? In particular, Maximize 0-Light is identical to Maximum Independent Set and therefore already extremely hard to approximate, so does Maximize W-Light become easier when W gets larger?

- What is the computational complexity of Maximize W-Light and Minimize $(W + 1)$-Heavy restricted to planar graphs when $W \in \{1, 2\}$?

- What is the computational complexity of Minimize W-Light and Maximize $(W + 1)$-Heavy for the special case $W = 1$?

- When G is a tree, all the problems are solvable in $O(n) = O(m)$ time by Corollary 2, Theorem 2, and Theorem 7. Although these methods are very simple, they achieve optimal running times since the size of the input is $\Omega(m)$. Do any other classes of graphs admit $O(m)$-time algorithms for some variants of the problems?

- Can Theorem 9 be strengthened to give non-trivial polynomial-time inapproximability bounds for Minimize W-Light and Maximize $(W + 1)$-Heavy?

- Is it possible to refine our polynomial-time approximation algorithms for Minimize W-Light and Maximize $(W + 1)$-Heavy to improve the approximation ratios in Theorems 10 and 11?

Finally, we briefly discuss two techniques from the literature that could lead to stronger results for special cases of the new problems.

- Baker [5] presented a *polynomial-time approximation scheme* (i.e., a polynomial-time $(1+\epsilon)$-approximation algorithm for any fixed $\epsilon > 0$) for Maximum Independent Set restricted to planar graphs. It first takes any planar embedding of the input graph G, partitions its vertices into successive layers $V_1, V_2, V_3, \ldots$, and defines a collection $\{G_1, G_2, \ldots, G_k\}$ of k-outerplanar graphs, where $k = \lceil 1 + (1/\epsilon) \rceil$, by removing the layers $V_i, V_{i+k}, V_{i+2k}, \ldots$ from a copy of G to get each G_i. It then runs an exact, polynomial-time algorithm for Maximum Independent Set restricted to k-outerplanar graphs on each G_i, and returns the best solution found. For any optimal solution I, at least one graph G_i contains at least $\frac{k-1}{k}$ of the vertices in I, so this method is a $(\frac{k}{k-1})$-approximation, i.e., a $(1+\epsilon)$-approximation. Baker's technique can be applied to many other optimization problems on planar graphs (see [5]), and if one could develop an exact, polynomial-time algorithm for Maximize W-Light with $W \in \{1, 2\}$ restricted to k-outerplanar graphs then the above analysis implies that this problem variant admits a polynomial-time approximation scheme for planar graphs as well.

- By formulating the problems in *monadic second-order logic* and applying Courcelle's theorem [13], one would obtain fixed-parameter tractable algorithms for graphs of bounded treewidth. Here, it is straightforward to come up with formulas of length $O(1)$ that express the condition that every edge in G is oriented (e.g., first assign an arbitrary ordering to the vertices in G and partition the edges into two sets, consisting of all edges oriented from a vertex with a smaller index to a vertex with a larger index and vice versa, and ensure that each edge belongs to exactly one of them) and that tell us in which direction any specified edge is oriented in an orientation. The difficulty is defining a compact formula, i.e., whose length does not depend on the size of the graph, for checking if a vertex has at most / at least W outgoing edges. We leave it as an open problem to resolve this issue.

Acknowledgments

The authors would like to thank Avraham Melkman, Manuel Sorge, and the anonymous reviewers for some insightful comments.

Addendum

Some additional results that extend the tables in Figure 2 have recently been obtained. They will appear in [2] and [31].

References

[1] Y. Asahiro, J. Jansson, E. Miyano, and H. Ono. Graph orientation to maximize the minimum weighted outdegree. *International Journal of Foundations of Computer Science*, 22(3):583–601, 2011. doi:10.1142/S0129054111008246.

[2] Y. Asahiro, J. Jansson, E. Miyano, and H. Ono. Degree-constrained graph orientation: Maximum satisfaction and minimum violation. *Theory of Computing Systems*, to appear. doi:10.1007/s00224-014-9565-5.

[3] Y. Asahiro, J. Jansson, E. Miyano, H. Ono, and K. Zenmyo. Approximation algorithms for the graph orientation minimizing the maximum weighted outdegree. *Journal of Combinatorial Optimization*, 22(1):78–96, 2011. doi:10.1007/s10878-009-9276-z.

[4] Y. Asahiro, E. Miyano, H. Ono, and K. Zenmyo. Graph orientation algorithms to minimize the maximum outdegree. *International Journal of Foundations of Computer Science*, 18(2):197–215, 2007. doi:10.1142/S0129054107004644.

[5] B. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994. doi:10.1145/174644.174650.

[6] B. Balasundaram, S. S. Chandramouli, and S. Trukhanov. Approximation algorithms for finding and partitioning unit-disk graphs into co-k-plexes. *Optimization Letters*, 4(3):311–320, 2010. doi:10.1007/s11590-009-0146-5.

[7] N. Bansal and M. Sviridenko. The Santa Claus problem. In *Proceedings of STOC 2006*, pages 31–40. ACM, 2006. doi:10.1145/1132516.1132522.

[8] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph Classes: A Survey*. SIAM Monographs on Discrete Mathematics and Applications, 1999. doi:10.1137/1.9780898719796.

[9] G. S. Brodal and R. Fagerberg. Dynamic representations of sparse graphs. In *Proceedings of WADS 1999*, volume 1663 of *Lecture Notes in Computer Science*, pages 342–351. Springer, 1999. doi:10.1007/3-540-48447-7_34.

[10] Z.-Z. Chen, M. Fellows, B. Fu, H. Jiang, Y. Liu, L. Wang, and B. Zhu. A linear kernel for Co-Path/Cycle Packing. In *Proceedings of AAIM 2010*, volume 6124 of *Lecture Notes in Computer Science*, pages 90–102. Springer, 2010. doi:10.1007/978-3-642-14355-7_10.

[11] M. Chrobak and D. Eppstein. Planar orientations with low out-degree and compaction of adjacency matrices. *Theoretical Computer Science*, 86(2):243–266, 1991. doi:10.1016/0304-3975(91)90020-3.

[12] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. The MIT Press, Massachusetts, 1990.

[13] B. Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

[14] A. Dessmark, K. Jansen, and A. Lingas. The maximum k-dependent and f-dependent set problem. In *Proceedings of ISAAC 1993*, volume 762 of *Lecture Notes in Computer Science*, pages 88–97. Springer, 1993. doi:10.1007/3-540-57568-5_238.

[15] I. Dinur and S. Safra. On the hardness of approximating minimum vertex cover. *Annals of Mathematics*, 162(1):439–485, 2005. doi:10.4007/annals.2005.162.439.

[16] T. Ebenlendr, M. Krčál, and J. Sgall. Graph balancing: A special case of scheduling unrelated parallel machines. *Algorithmica*, 68(1):62–80, 2014. doi:10.1007/s00453-012-9668-9.

[17] U. Feige. Approximating maximum clique by removing subgraphs. *SIAM Journal on Discrete Mathematics*, 18(2):219–225, 2004. doi:10.1137/S089548010240415X.

[18] M. R. Fellows, J. Guo, H. Moser, and R. Niedermeier. A generalization of Nemhauser and Trotter's local optimization theorem. *Journal of Computer and System Sciences*, 77(6):1141–1158, 2011. doi:10.1016/j.jcss.2010.12.001.

[19] T. Fujito. A unified approximation algorithm for node-deletion problems. *Discrete Applied Mathematics*, 86(2–3):213–231, 1998. doi:10.1016/S0166-218X(98)00035-3.

[20] H. N. Gabow. Upper degree-constrained partial orientations. In *Proceedings of SODA 2006*, pages 554–563. SIAM, 2006.

[21] H. N. Gabow and R. E. Tarjan. A linear-time algorithm for finding a minimum spanning pseudoforest. *Information Processing Letters*, 27(5):259–263, 1988. doi:10.1016/0020-0190(88)90089-0.

[22] M. Garey and D. Johnson. *Computers and Intractability – A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York, 1979.

[23] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972. doi:10.1137/0201013.

[24] F. Gavril. Testing for equality between maximum matching and minimum node covering. *Information Processing Letters*, 6(6):199–202, 1977. doi:10.1016/0020-0190(77)90068-0.

[25] A. V. Goldberg and S. Rao. Beyond the flow decomposition barrier. *Journal of the ACM*, 45(5):783–797, 1998. doi:10.1145/290179.290181.

[26] M. M. Halldórsson. Approximations of weighted independent set and hereditary subset problems. *Journal of Graph Algorithms and Applications*, 4(1):1–16, 2000. doi:10.7155/jgaa.00020.

[27] F. Havet, R. J. Kang, and J.-S. Sereni. Improper coloring of unit disk graphs. *Networks*, 54(3):150–164, 2009. doi:10.1002/net.20318.

[28] L. S. Heath and J. P. C. Vergara. Edge-packing in planar graphs. *Theory of Computing Systems*, 31(6):629–662, 1998. doi:10.1007/s002240000107.

[29] G. Karakostas. A better approximation ratio for the vertex cover problem. *ACM Transactions on Algorithms*, 5(4), 2009. Article 41. doi:10.1145/1597036.1597045.

[30] R. M. Karp. Reducibility among combinatorial problems. In *Proceedings of Complexity of Computer Computations*, pages 85–103. The IBM Research Symposia Series, Plenum Press, 1972. doi:10.1007/978-1-4684-2001-2_9.

[31] K. Khoshkhah. On finding orientations with the fewest number of vertices with small out-degree. *Discrete Applied Mathematics*, 194:163 – 166, 2015. doi:10.1016/j.dam.2015.05.007.

[32] C. Komusiewicz, F. Hüffner, H. Moser, and R. Niedermeier. Isolation concepts for efficiently enumerating dense subgraphs. *Theoretical Computer Science*, 410(38–40):3640–3654, 2009. doi:doi:10.1016/j.tcs.2009.04.021.

[33] B. McClosky. *Independence Systems and Stable Set Relaxations*. PhD thesis, Rice University, U.S.A., 2008.

[34] H. Moser, R. Niedermeier, and M. Sorge. Exact combinatorial algorithms and experiments for finding maximum k-plexes. *Journal of Combinatorial Optimization*, 24(3):347–373, 2012. doi:10.1007/s10878-011-9391-5.

[35] C. Nash-Williams. Decomposition of finite graphs into forests. *Journal of the London Mathematical Society*, s1-39(1):12, 1964. doi:10.1112/jlms/s1-39.1.12.

[36] N. Nishimura, P. Ragde, and D. M. Thilikos. Fast fixed-parameter tractable algorithms for nontrivial generalizations of vertex cover. *Discrete Applied Mathematics*, 152(1–3):229–245, 2005. doi:10.1016/j.dam.2005.02.029.

[37] A. Schrijver. *Combinatorial Optimization*. Springer, 2003.

[38] V. Venkateswaran. Minimizing maximum indegree. *Discrete Applied Mathematics*, 143(1–3):374–378, 2004. doi:10.1016/j.dam.2003.07.007.

[39] D. Zuckerman. Linear degree extractors and the inapproximability of Max Clique and Chromatic Number. *Theory of Computing*, 3(1):103–128, 2007. doi:10.4086/toc.2007.v003a006.