Primal-Dual Approximation Algorithms for Node-Weighted Network Design in Planar Graphs*

Piotr Berman and Grigory Yaroslavtsev

Pennsylvania State University, USA
{berman,grigory}@cse.psu.edu

Abstract. We present primal-dual algorithms which give a 2.4 approximation for a class of node-weighted network design problems in planar graphs, introduced by Demaine, Hajiaghayi and Klein (ICALP'09). This class includes Node-Weighted Steiner Forest problem studied recently by Moldenhauer (ICALP'11) and other node-weighted problems in planar graphs that can be expressed using $(0, 1)$-proper functions introduced by Goemans and Williamson. We show that these problems can be equivalently formulated as feedback vertex set problems and analyze approximation factors guaranteed by different violation oracles within the primal-dual framework developed by Goemans and Williamson.

1 Introduction

In feedback vertex set problems the input is a graph $G = (V, E)$, a family of cycles $\mathcal{C}$ in G and a function $w\colon V \to \mathbb{R}^{\geq 0}$. The goal is to find a set of vertices $H \subset V$ which contains a node in every cycle in $\mathcal{C}$ such that the total weight of vertices in H is minimized. This is a special case of the hitting set problem, where sets correspond to the cycles of $\mathcal{C}$. There five natural examples for the family $\mathcal{C}$.

- All cycles. This is Feedback Vertex Set problem (FVS).
- Odd cycles. If $H \subset V$ is a hitting set for all odd-length cycles then the subgraph of G, induced by the vertex set $V \setminus H$ is bipartite. This is Bipartization problem (BIP).
- The set of all cycles which contain at least one node from a given set of nodes. This is Subset Feedback Vertex Set problem (S-FVS).
- The set of all directed cycles of a given directed graph. This is Directed Feedback Vertex Set problem (D-FVS).
- In Node-Weighted Steiner Forest problem we are given a weighted graph and a set of terminal pairs (s_i, t_i). The goal is to select $S \subset V$ such that in the subgraph induced by S all terminal pairs are connected. In Section 2.1 we show that Node-Weighted Steiner Forest belongs to a class of problems which can be expresed as a hitting set problem for an appropriately defined collection of cycles.

* G.Y. is supported by NSF / CCF CAREER award 0845701 and by College of Engineering Fellowship.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 50–60, 2012.

Table 1. Planar graphs

Problem	Previous work (our analysis)[1]	Our work	Hardness
FVS	10 [3], 3 (18/7) [11], 2 [4,1]		
BIP, D-FVS, S-FVS	3 (18/7) [11]	2.4	NP-hard [20]
NODE-WEIGHTED STEINER FOREST	6 [5], 3 (18/7) [17]		

While in general graphs FVS can be approximated within factor of 2 for all graphs, as shown by Becker and Geiger [4] and Bafna, Berman and Fujito [1], hitting a restricted family of cycles can be much harder. For example, the best known approximation ratio for graph bipartization in general graphs is $O(\log n)$ by Garg, Vazirani and Yannakakis [9]. For D-FVS the best known approximation is $O(\log n \log\log n)$, as shown by Even, Naor, Schieber and Sudan [8]. These and other results for general graphs are discussed in the full version.

Yannakakis [20] has given an NP-hardness proof for many vertex deletion problems restricted to planar graphs which applies to all problems that we consider. For planar graphs, the unweighted FEEDBACK VERTEX SET problem admits a PTAS, as shown by Demaine and Hajiaghayi [6] using a bidimensionality technique. Goemans and Williamson [11] created a framework for primal-dual algorithms that for planar instances of all above problems provide approximation algorithms with constant approximation factors. More specifically, they showed 9/4-approximations for FVS, S-FVS, D-FVS and BIP. For NODE-WEIGHTED STEINER FOREST it was shown by Demaine, Hajiaghayi and Klein [5] that the generic framework of Goemans and Williamson gives a 6-approximation which was improved to 9/4-approximation by Moldenhauer [17]. However, the original paper by Goemans and Williamson [11] contains a mistake in the analysis. Similar mistake was repeated in [17]. We exhibit the mistake on an example and prove that no worse example exists. More precisely, primal-dual approximation algorithms of Goemans and Williamson for all problems described above give approximation factor 18/7 rather than 9/4. We also give an improved version of the violation oracle which can be used within the primal-dual framework of Goemans and Williamson and guarantees approximation factor 2.4. Results for planar graphs are summarized in Table 1.

Applications and ramifications. Node-weighted Steiner problems have been studied theoretically in many different settings, see e.g. [15,18,19,16]. Applications of such problems range from maintenance of electric power networks [12] to computational sustainability [7]. Experimental evaluation of primal-dual algorithms for feedback vertex set problems in planar graphs in applications to VLSI design has been done by Kahng, Vaya and Zelikovsky [13].

Organization. We give basic definitions and preliminary observations in Section 2. In Section 2.1 we show that a wide class of node-weighted network design problems in planar graphs, introduced by Demaine, Hajiaghayi and Klein [5], can

[1] See discussion in the text.

be equivalently defined as a class of hitting set problems for appropriately defined collections of cycles satisfying *uncrossing property*, as introduced by Goemans and Williamson [11]. In Section 3 we introduce local-ratio analog of primal-dual framework of Goemans and Williamson for such problems and give examples of violation oracles which can be used within this framework.

In Section 4 we give corrected analysis of the approximation factor achieved by the generic primal-dual algorithm with a violation oracle, presented by Goemans and Williamson in [11]. In the full version we present analysis of primal-dual algorithms with a new violation oracle which gives approximation factor 2.4. In Section 4.2 we show examples, on which these approximation factors are achieved.

2 Preliminaries

A *simple* cycle of length k is a sequence of vertices $v_1, \ldots, v_{k+1}$, where $v_{k+1} \equiv v_1$, all vertices $v_1, \ldots v_k$ are distinct, $(v_i, v_{i+1}) \in E$ for all $1 \leq i \leq k$ and all these edges are distinct. Note that in undirected simple graphs a simple cycle has length at least three. For a cycle C, the edge set of C is denoted as $E(C)$, although to simplify presentation we may refer to it as just C.

Every planar graph has a combinatorial embedding which for every vertex specifies a cyclic ordering of edges that are adjacent to it. A subset $U \subset V$ defines $G[U]$, the *induced subgraph* of G, with node set U and edges $\{(u,v) \in E : \; u, v \in U\}$. An embedding of a planar graph naturally defines embeddings of all its induced subgraphs. We denote the set of faces of a planar graph as F (for a standard definition of the set of faces via a combinatorial embedding, see e.g. [14]). The *planar dual* of a graph G is graph $G^* = (F, E')$ where F is the set of faces of G, and E' is the set of pairs of faces that share an edge. We select one face F_0 as the *outer face*.

For a simple cycle $C = (v_1, \ldots, v_{k+1})$ we denote the set of faces that are surrounded by C as $Faces(C)$. More formally, let E'' be the set of pairs of faces that share an edge that is not on C then in (F, E'') has exactly two connected components. We denote as $Faces(C)$ the connected component of (F, E'') that does not contain the outer face F_0.

For a weight function $w : V \to \mathbb{R}$ and a set $S \subseteq V$ we denote $w(S) = \sum_{e \in S} w(e)$.

2.1 Uncrossable Families of Cycles and Proper Functions

Two simple cycles C, D are *crossing* if neither $Faces(C) \subset Faces(D)$, nor $Faces(D) \subset Faces(C)$, nor $Faces(D) \cap Faces(C) = \varnothing$. A family of simple cycles $\mathcal{Z}$ is *laminar* iff it does not contain a pair of crossing cycles.

Our algorithms apply to every family of cycles that satisfies the following (similar to the *uncrossing property* of [11]). If two simple cycles C_1, C_2 are crossing then there exist paths $P_1 \subseteq C_1$ and $P_2 \subseteq C_2$, such that P_1 (P_2) intersects C_2 (C_1) only at its endpoints and P_2 contains an edge in the interior of C_1.

Definition 2.1 (Uncrossing property [11]). *A family of simple cycles $\mathcal{C}$ has the uncrossing property if for every pair of crossing cycles $C_1, C_2 \in \mathcal{C}$ as described above either $P_1 \cup P_2 \in \mathcal{C}$ and $(C_1 \setminus P_1) \cup (C_2 \setminus P_2)$ contains a cycle in $\mathcal{C}$, or $(C_1 \setminus P_1) \cup P_2 \in \mathcal{C}$ and $(C_2 \setminus P_2) \cup P_1$ contains a cycle in $\mathcal{C}$.*

Many natural families of cycles satisfy the *uncrossing property*. Goemans and Williamson [11] showed this for FVS, D-FVS, BIP, and S-FVS. We show that these problems belong to a wider class of node-weighted connectivity problems in planar graphs which can be expressed as problems of finding hitting sets for families of cycles satisfying the uncrossing property. To state it formally we introduce some definitions.

Definition 2.2 ((0, 1)-proper function). *A Boolean function $f: 2^V \to \{0,1\}$ is* proper *if $f(\emptyset) = 0$ and it satisfies the following properties:*

1. *(Symmetry) $f(S) = f(V \setminus S)$.*
2. *(Disjointness) If $S_1 \cap S_2 = \emptyset$ and $f(S_1) = f(S_2) = 0$ then $f(S_1 \cup S_2) = 0$.*

These properties imply the property known as *complementarity*: if $A \subseteq S$ and $f(S) = f(A) = 0$ then $f(S \setminus A) = 0$.

For a set $S \subseteq V$, let $\Gamma(S)$ be its boundary, i.e. the set of nodes not in S which have a neighbor in S, or formally $\Gamma(S) = \{v \in V | v \notin S, \exists u \in S: (u, v) \in E\}$. As observed by Demaine, Hajiaghayi and Klein [5], a wide class of node-weighted network design problems can be formulated as the following generic integer program, where $f: 2^V \to \{0,1\}$ is a (0, 1)-proper function:

$$\text{Minimize: } \sum_{v \in V} w(v)x(v) \tag{1}$$

$$\text{Subject to: } \sum_{v \in \Gamma(S)} x(v) \geq f(S) \qquad \text{for all } S \subseteq V \tag{2}$$

$$x(v) \in \{0,1\} \qquad \text{for all } v \in V, \tag{3}$$

For example, for Node-Weighted Steiner Forest the corresponding (0, 1)-proper function is defined as follows: $f(S) = 1$ iff there exists a pair of terminals (s_i, t_i), such that $|S \cap \{s_i, t_i\}| = 1$. The edge-weighted version of this program was introduced by Goemans and Williamson in [10]. Note that without loss of generality we can assume that the input graph is triangulated. Otherwise we add extra nodes of infinite cost inside each face and connect these new nodes to all nodes on their faces without changing the cost of the optimum solution. Let V' be the set of nodes after such extension. Then the corresponding (0, 1)-proper function f' for the extended instance is defined for all $S \subseteq V'$ as $f'(S) = f(S \cap V)$.

In Theorem 2.1 we show that a problem expressed by an integer program (1-3) with some (0, 1)-proper function f can also be expressed as a problem of hitting a collection of cycles with the uncrossing property. We give some definitions and simplifying assumptions first.

Definition 2.3 (Active sets and boundaries). *Assume that* $f\colon 2^V \to \{0,1\}$ *is a* $(0,1)$*-proper function. If* $f(S) = 1$ *we say that* S *is* active, *and that* $\Gamma(S)$ *is an* active boundary. *If* $\Gamma(S)$ *is a simple cycle we call it an* active simple boundary. *We denote the collection of all active simple boundaries as* $\mathcal{C}^f$.

Using this terminology the integer program (1-3) expresses the problem of finding a minimum weight hitting set for the collection of all active boundaries. Note that every active singleton set $\{s\}$ must be included in the solution because $\{s\} = \Gamma(V \setminus \{s\})$ and $V \setminus \{s\}$ is active by symmetry, so $\{s\}$ has to be hit. Let S_0 be the set of such singletons. Using the observation above we can simplify the integer program (1-3) by using only inequalities of type (2) such that $\Gamma(S) \cap S_0 = \varnothing$. By disjointness of f, if $\Gamma(S) \cap S_0 = \varnothing$ then $f(\Gamma(S)) = 0$, *i.e.* every active boundary in the inequalities (2) of the simplified program is inactive.

In Lemma 2.1 we show that hitting all active boundaries is equivalent to hitting $\mathcal{C}^f$ because every active boundary contains an active simple boundary as a subset. This lemma is proved in the full version.

Lemma 2.1. *Let* $G(V,E)$ *be a connected triangulated planar graph,* f *be a* $(0,1)$*-proper function and* $\Gamma \subset V$ *be a set with the following properties:*

1. $f(\{a\}) = 0$ *for every* $a \in \Gamma$.
2. $f(B) = 1$ *for some* B *that is a connected component of* $V \setminus \Gamma$.

Then every set C *which is a minimal subset of* Γ *satisfying the two properties above is a simple cycle.*

Then we show that the family of active simple boundaries $\mathcal{C}^f$ satisfies the uncrossing property.

Theorem 2.1. *Let* $G(V,E)$ *be a triangulated planar graph. For every* $(0,1)$*-proper function* $f\colon 2^V \to \{0,1\}$ *the collection of active simple boundaries* $\mathcal{C}^f$ *forms an uncrossable family of cycles.*

Proof. Consider two active simple boundaries $\Gamma(S_1)$ and $\Gamma(S_2)$. If $\Gamma(S_2)$ crosses $\Gamma(S_1)$ then there exists a collection of edge-disjoint paths in $\Gamma(S_2)$ which we denote as P, such that each path $P_i \in P$ has only two nodes in common with $\Gamma(S_1)$. Each path $P_i \in P$ partitions $S_1 \setminus P_i$ into two parts which we denote as A_i^1 and A_i^2 respectively. Let's fix a path $P_i \in P$, such that at A_i^1 doesn't contain any other paths from P.

There are two cases: $A_i^1 \cap S_2 = \emptyset$ and $A_i^1 \subseteq S_2$. They are symmetric because if $A_i^1 \subseteq S_2$ we can replace the set S_2 by a set $S_2' = V \setminus S_2 \setminus \Gamma(S_2)$, ensuring that $A_i^1 \cap S_2 = \emptyset$. Note that the boundary doesn't change after such replacement, because $\Gamma(S_2) = \Gamma(S_2')$. By symmetry of f we have that $f(S_2) = f(V \setminus S_2) = 1$. Because $f(\Gamma(S_2)) = 0$ by disjointness we have $f(V \setminus S_2 \setminus \Gamma(S_2)) = f(S_2') = 1$, so S_2' is also an active set.

This is why it is sufficient to consider only the case when $A_i^1 \cap S_2 = \emptyset$. We will show the following auxiliary lemma:

Lemma 2.2. *Let $A_1, A, B \subseteq V$ be such that $A_1 \subseteq A$, $A_1 \cap B = \emptyset$ and $f(A) = f(B) = 1$. Then at least one of the following two statements holds:*
1. $f(A_1 \cup B) = f(A \setminus A_1) = 1$.
2. $f(A_1) = \max\left[f(B \setminus (A \setminus A_1)), f((A \setminus A_1) \setminus B)\right] = 1$.

The proof of the lemma follows from the properties of $(0,1)$-proper functions and is given in the full version.

To show the uncrossing property for cycles $C_1 = \Gamma(S_1)$ and $C_2 = \Gamma(S_2)$ we select the paths in the definition of the uncrossing property as $P_1 = \Gamma(A_i^2) \setminus P_i$ and $P_2 = P_i$. Now we can apply Lemma 2.2 to sets A_i^1, S_1 and S_2, because $A_i^1 \subseteq S_1$, $A_i^1 \cap S_2 = \emptyset$ and $f(S_1) = f(S_2) = 1$. Thus, by Lemma 2.2 either $f(A_i^1 \cup S_2) = f(S_1 \setminus A_i^1) = 1$ or $f(A_i^1) = max(f(S_2 \setminus (S_1 \setminus A_i^1)), f((S_1 \setminus A_i^1) \setminus S_2)) = 1$. In the first case we have $f(A_i^2) = f(A_i^1 \cup S_2) = 1$ and thus both cycles $P_1 \cup P_2 = \Gamma(A_i^2)$ and $(C_1 \setminus P_1) \cup (C_2 \setminus P_2) = \Gamma(A_i^1 \cup S_2)$ are active simple boundaries. In the second case $f(A_i^1) = 1$ and thus the cycle $(C_1 \setminus P_1) \cup P_2 = \Gamma(A_1)$ is an active simple boundary. The cycle $(C_2 \setminus P_2) \cup P_1$ is not necessarily simple, but it forms a boundary of an active set $(S_2 \setminus (S_1 \setminus A_i^1)) \cup ((S_1 \setminus A_i^1) \setminus S_2)$. Thus, by Lemma 2.1 it contains an active simple boundary, which is a cycle in $\mathcal{C}^f$.

3 Algorithm

3.1 Generic Local-Ratio Algorithm

We will use a local-ratio analog of a generic primal-dual algorithm formulated by Goemans and Williamson [11] which we state as Algorithm 1. As observed in the full verison of [17] these two formulations are equivalent for the problems that we consider (see also [2]).

Algorithm 1: Generic local-ratio algorithm $(G(V,E), w, \mathcal{C})$

1 $\bar{w} \leftarrow w$.
2 $S \leftarrow \{u \in V : \bar{w}(u) = 0\}$.
3 **while** *S is not a hitting set for $\mathcal{C}$* **do**
4 $\quad \mathcal{M} \leftarrow$ Violation$(G, \mathcal{C}, S)$.
5 $\quad c_{\mathcal{M}}(u) \leftarrow |\{M \in \mathcal{M} : u \in M\}|$, for all $u \in V \setminus S$.
6 $\quad \alpha \leftarrow \min_{u \in V \setminus S} \frac{\bar{w}(u)}{c_{\mathcal{M}}(u)}$.
7 $\quad \bar{w}(u) \leftarrow \bar{w}(u) - \alpha c_{\mathcal{M}}(u)$, for all $u \in V \setminus S$.
8 $\quad S \leftarrow \{u \in V : \bar{w}(u) = 0\}$.
end
9 **return** a minimal hitting set $H \subset S$ of $\mathcal{C}$.

We say that a hitting set for a collection of cycles is minimal, if it doesn't contain another hitting set as its proper subset. Note that we don't need to specify the collection of cycles $\mathcal{C}$ explicitly. Instead the generic algorithm requires that we specify an oracle Violation$(G, \mathcal{C}, S)$ used in Step 4. Given a graph G, collection of cycles $\mathcal{C}$ and a solution S if there are some cycles in $\mathcal{C}$ which are not hit by S this oracle should return a non-empty collection of such cycles, otherwise it should return the empty set. Such an oracle also allows to perform Step 3 and Step 9 without explicitly specifying $\mathcal{C}$.

The performance guarantee of the generic algorithm depends on the oracle used as described below.

Theorem 3.1 (Local-ratio analog of Theorem 3.1 in [11]). *If the set $\mathcal{M}$ returned by a violation oracle used in Step 4 of the generic local-ratio Algorithm 1 satisfies that for any minimal solution $\breve{H}$:*

$$c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|,$$

then Algorithm 1 returns a hitting set H of cost $w(H) \leq \gamma w(H^*)$, where H^* is the optimum solution.

We give the proof of this theorem for completeness in the full version.

The simplest violation oracles return a single cycle. Bar-Yehuda, Geiger, Naor and Roth [3] show that for FVS this approach can give a 10-approximation for planar graphs and Goemans and Willamson [11] improve it to a 5-approximation. They also analyzed an oracle, which returns a collection of all faces in $\mathcal{C}$, which are not hit by the current solution, and showed such oracle gives a 3-approximation for all families of cycles satisfying uncrossing property. Thus, by Theorem 2.1 such oracle gives a 3-approximation for all problems that we consider. We now give more complicated examples of violation oracles which give better approximation factors.

3.2 Face Minimal Violation Oracles

Definition 3.1. *Given $S \subset V$, $\mathcal{C}(S) = \{C \in \mathcal{C} : \ C \cap S = \varnothing\}$. A cycle $C \in \mathcal{C}(S)$ is* face minimal *if there is no $D \in \mathcal{C}(S)$ such that $Faces(D) \subsetneq Faces(C)$.* Minimal$(S) = \{C \in \mathcal{C}(S) : C$ *is face minimal*$\}$.

Goemans and Williamson [11] showed that using Minimal(S) as Violation $(G, \mathcal{C}, S)$ leads to approximation ratio 3. Other violation oracles we discuss can be computed by selecting a subset of Minimal(S). Thus the algorithms we discuss run in polynomial time if the function Minimal(S) can be computed in polynomial time. This is shown in [11,17] for the problems considered there. This also holds in general for sets of cycles defined by $(0,1)$-proper functions.

Lemma 3.1. *For a family of cycles $\mathcal{C}^f$ defined by a $(0,1)$-proper function* Minimal(S) *can be computed in polynomial time.*

We give a sketch of the proof below. Let $\mathcal{A}$ be the set of active connected components of $V \setminus S$. Each cycle in Minimal(S) will be a minimal subset of $\Gamma(A)$ for some $A \in \mathcal{A}$. However, we need to show how to find all cycles of Minimal(S) rather than one.

We start by defining a partial order on $\mathcal{A}$. For a fixed $A \in \mathcal{A}$ we have set $\mathcal{K}(A)$ of connected components of $V \setminus \Gamma(A)$; note that $A \in \mathcal{K}(A)$. We say that a $B \in \mathcal{K}(A)$ is an outer (inner) component if B contains at least one node of the outer face (does not contain any). One can see that there exists at most one outer component in $\mathcal{K}(A)$. We say that A dominates $A' \in \mathcal{A}$ if some B is an inner component of $\mathcal{K}(A)$, $B \neq A$ and $A' \subset B$. This relation is anti-symmetric and transitive, hence it defines a partial order. We can show that each cycle of Minimal(S) is contained in $\Gamma(A)$ where A is a minimal element of $\mathcal{A}$ in terms of domination.

Then given such minimal A we first insert to A those nodes from $\Gamma(A)$ that have neighbors only in $\Gamma(A) \cup A$. Then we can show that resulting smaller $\Gamma(A)$ induces a subgraph that can be uniquely decomposed into a family of simple cycles, and exactly one of those cycles is a boundary of an active set. Details will be provided in the full version.

3.3 Minimal Pocket Violation Oracles

The following oracle, introduced by Goemans and Williamson [11], returns a collection of faces in $\mathcal{C}$ inside a minimal *pocket* not hit by the current solution H.

Definition 3.2. *A* pocket *for a planar graph $G(V, E)$ and a cycle collection $\mathcal{C}$ is a set $U \subseteq V$ such that:*

1. *The set U contains at most two nodes with neighbors outside U.*
2. *The induced subgraph $G[U]$ contains at least one cycle in $\mathcal{C}$.*

Algorithm 2: Minimal-Pocket-Violation $(G, \mathcal{C}, S)$

$\mathcal{C}_0 \leftarrow \{c \in \mathcal{C} : c \text{ not hit by } S\}$
$\mathcal{M} \leftarrow$ Minimal(S)
Construct a graph G_S by removing from G:
  All edges in the interior of cycles of $\mathcal{M}$.
  All vertices which are not adjacent to any edges.
Let U_0 be a pocket for G_S and $\mathcal{C}_0$ which doesn't contain any other pockets.
return A collection of all cycles in $\mathcal{C}_0$ which are faces of $G_S[U_0]$.

As in the generic algorithm, we will not specify $\mathcal{C}$ and $\mathcal{C}_0$ explicitly, but will rather use an oracle to check relevant properties with respect to them. We show analysis of the approximation factor obtained with this oracle in Section 4.

We will obtain a better approximation ratio by analyzing the following oracle in the full version.

Definition 3.3. *A* triple pocket *for a planar graph $G(V, E)$ and a cycle collection $\mathcal{C}$ is a set $U \subseteq V$ such that:*

1. *The set U contains at most three nodes with neighbors outside U.*
2. *The induced subgraph $G_S[U]$ has at least three faces in $\mathcal{C}$.*

The violation oracle Minimal-3-Pocket-Violation finds a minimal U_0 that is either a pocket or a triple pocket, and otherwise works like Minimal-Pocket-Violation.

4 18/7 Approximation Ratio with Pocket Oracle

According to Theorem 3.1, to show that Algorithm 1 with Minimal-Pocket-Violation oracle has approximation factor 18/7 it suffices to prove the following:

Theorem 4.1. *In every iteration of the generic local-ratio algorithm (Algorithm 1) with oracle* Minimal-Pocket-Violation *for every minimal hitting set* $\breve{H}$ *of* $\mathcal{C}$ *we have* $c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|$ *for* $\gamma = 18/7$.

The proof is in the full version.

4.1 12/5 Approximation Ratio with Triple Pocket Oracle

In the generic local-ratio algorithm we can change the implementation of the oracle Violation. Namely we can use Minimal-3-Pocket-Violation which in turn is a modification of Algorithm 2: in line 6 select U_0 as a minimal triple pocket. Note that a triple pocket is defined by three (or less) nodes that form $\Gamma(V - U_0)$, hence we still have a polynomial time. In the full version we prove

Theorem 4.2. *In every iteration of the generic local-ratio algorithm (Algorithm 1) with oracle* Minimal-3-Pocket-Violation *for every minimal hitting set* $\breve{H}$ *of* $\mathcal{C}$ *we have* $c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|$ *for* $\gamma = 12/5$.

4.2 Tight Examples

We show instances of graphs, on which the primal-dual algorithm with oracles Minimal-Pocket-Violation and Minimal-3-Pocket-Violation gives 18/7 and 12/5 approximations respectively.

Our examples are for the Subset Feedback Vertex Set problem. Recall that in this problem we need to hit all cycles which contain a specified set of "special" nodes. Our examples are graphs with no *pockets* (or triple pockets), in which every face belongs to the family of cycles that we need to hit – this is ensured by selection of "special" nodes, which are marked with a star . The weights of vertices are assigned as follows. Given a node u with degree $d(u)$, its weight is $w(u) = d(u)$ if u is a solid dot and $w(u) = d(u) + \epsilon$ otherwise (for some negligibly small value of ϵ).

First we show an example for the oracle Minimal-Pocket-Violation in Figure 1. Because there are no pockets, the first execution of the violation oracle returns the collection of all faces in the graph. Thus, in each building block of Picture 1 (which shows 5 such blocks from left to right), the primal-dual algorithm selects the black dots with total weight 18 while stars also form a valid solution with weight $7 + 3\epsilon$. Hence the ratio will be arbitrarily close to 18/7, if we repeat the building block many times.

Similar family of examples for the oracle Minimal-3-Pocket-Violation is shown in Figure 2. In these examples there are no pockets or triple pockets, so the oracle Minimal-3-Pocket-Violation returns the collection of all faces in the graph. As above, the primal-dual algorithm selects the black dots with total weight 12 within each block, while the cost of the solution given by the stars is $5 + 2\epsilon$, so we can make the ratio arbitrarily close to 12/5.

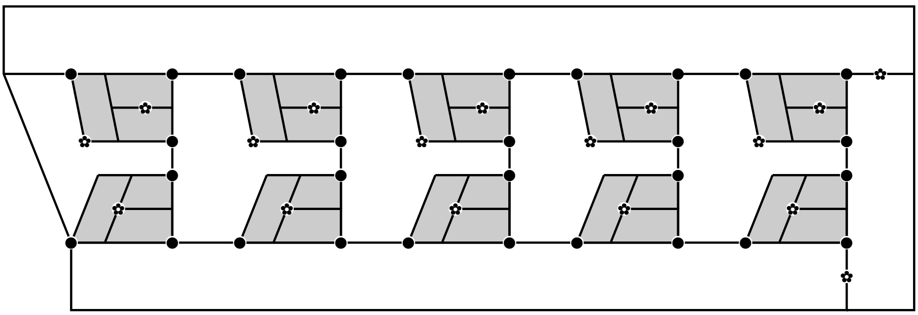

Fig. 1. Family of instances of S-FVS with approximation factor 18/7 for the primal-dual algorithm with oracle Minimal-Pocket-Violation

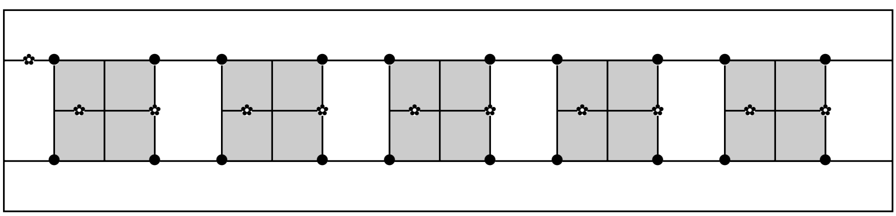

Fig. 2. Family of instances of S-FVS with approximation factor 12/5 for primal-dual algorithm with oracle Minimal-3-Pocket-Violation

References

1. Bafna, V., Berman, P., Fujito, T.: A 2-approximation algorithm for the undirected feedback vertex set problem. SIAM J. Discrete Math. 12(3), 289–297 (1999)
2. Bar-Yehuda, R., Bendel, K., Freund, A., Rawitz, D.: Local ratio: A unified framework for approxmation algrithms in memoriam: Shimon even 1935-2004. ACM Comput. Surv. 36(4), 422–463 (2004)
3. Bar-Yehuda, R., Geiger, D., Naor, J.S., Roth, R.M.: Approximation algorithms for the vertex feedback set problem with applications to constraint satisfaction and bayesian inference. In: SODA 1994, pp. 344–354. SIAM, Philadelphia (1994), `http://dl.acm.org/citation.cfm?id=314464.314514`

4. Becker, A., Geiger, D.: Optimization of pearl's method of conditioning and greedy-like approximation algorithms for the vertex feedback set problem. Artif. Intell. 83(1), 167–188 (1996)
5. Demaine, E.D., Hajiaghayi, M., Klein, P.N.: Node-Weighted Steiner Tree and Group Steiner Tree in Planar Graphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5555, pp. 328–340. Springer, Heidelberg (2009)
6. Demaine, E.D., Hajiaghayi, M.: Bidimensionality: new connections between fpt algorithms and ptass. In: SODA 2005, pp. 590–601. SIAM, Philadelphia (2005), `http://dl.acm.org/citation.cfm?id=1070432.1070514`
7. Dilkina, B., Gomes, C.P.: Solving Connected Subgraph Problems in Wildlife Conservation. In: Lodi, A., Milano, M., Toth, P. (eds.) CPAIOR 2010. LNCS, vol. 6140, pp. 102–116. Springer, Heidelberg (2010)
8. Even, G., (Seffi) Naor, J., Schieber, B., Sudan, M.: Approximating minimum feedback sets and multicuts in directed graphs. Algorithmica 20, 151–174 (1998)
9. Garg, N., Vazirani, V.V., Yannakakis, M.: Approximate max-flow min-(multi)cut theorems and their applications. SIAM J. Comput. 25, 235–251 (1996)
10. Goemans, M.X., Williamson, D.P.: A general approximation technique for constrained forest problems. SIAM J. Comput. 24(2), 296–317 (1995)
11. Goemans, M.X., Williamson, D.P.: Primal-dual approximation algorithms for feedback problems in planar graphs. Combinatorica 18, 37–59 (1998)
12. Guha, S., Moss, A., Naor, J., Schieber, B.: Efficient recovery from power outage (extended abstract). In: STOC 1999, pp. 574–582 (1999)
13. Kahng, A.B., Vaya, S., Zelikovsky, A.: New graph bipartizations for double-exposure, bright field alternating phase-shift mask layout. In: ASP-DAC 2001, pp. 133–138. ACM, New York (2001)
14. Klein, P.: Optimization Algorithms for Planar Graphs, `http://www.planarity.org/`
15. Klein, P.N., Ravi, R.: A nearly best-possible approximation algorithm for node-weighted steiner trees. J. Algorithms 19(1), 104–115 (1995)
16. Li, X., Xu, X.-H., Zou, F., Du, H., Wan, P., Wang, Y., Wu, W.: A PTAS for Node-Weighted Steiner Tree in Unit Disk Graphs. In: Du, D.-Z., Hu, X., Pardalos, P.M. (eds.) COCOA 2009. LNCS, vol. 5573, pp. 36–48. Springer, Heidelberg (2009)
17. Moldenhauer, C.: Primal-Dual Approximation Algorithms for Node-Weighted Steiner Forest on Planar Graphs. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011. LNCS, vol. 6755, pp. 748–759. Springer, Heidelberg (2011)
18. Moss, A., Rabani, Y.: Approximation algorithms for constrained node weighted steiner tree problems. SIAM J. Comput. 37(2), 460–481 (2007)
19. Remy, J., Steger, A.: Approximation Schemes for Node-Weighted Geometric Steiner Tree Problems. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX and RANDOM 2005. LNCS, vol. 3624, pp. 221–232. Springer, Heidelberg (2005)
20. Yannakakis, M.: Node-and edge-deletion np-complete problems. In: STOC 1978, pp. 253–264. ACM, New York (1978)