Minimum Leaf Out-Branching Problems

Gregory Gutin[1], Igor Razgon[2], and Eun Jung Kim[1]

[1] Department of Computer Science
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
gutin(eunjung)@cs.rhul.ac.uk
[2] Department of Computer Science
University College Cork, Ireland
i.razgon@cs.ucc.ie

Abstract. Given a digraph D, the Minimum Leaf Out-Branching problem (MinLOB) is the problem of finding in D an out-branching with the minimum possible number of leaves, i.e., vertices of out-degree 0. We prove that MinLOB is polynomial-time solvable for acyclic digraphs. In general, MinLOB is NP-hard and we consider three parameterizations of MinLOB. We prove that two of them are NP-complete for every value of the parameter, but the third one is fixed-parameter tractable (FPT). The FPT parametrization is as follows: given a digraph D of order n and a positive integral parameter k, check whether D contains an out-branching with at most $n-k$ leaves (and find such an out-branching if it exists). We find a problem kernel of order $O(k \cdot 2^k)$ and construct an algorithm of running time $O(2^{O(k \log k)} + n^3)$, which is an 'additive' FPT algorithm.

1 Introduction

We say that a subgraph T of a digraph D is an *out-tree* if T is an oriented tree with only one vertex s of in-degree zero (called *the root*). The vertices of T of out-degree zero are called *leaves*. If T is a spanning out-tree, i.e. $V(T) = V(D)$, then T is called an *out-branching* of D. Given a digraph D, the *Minimum Leaf Out-Branching* problem (*MinLOB*) is the problem of finding an out-branching with the minimum possible number of leaves in D. Denote this minimum by $\ell_{\min}(D)$. When D has no out-branching, we write $\ell_{\min}(D) = 0$. Notice that not every digraph D has an out-branching. It is not difficult to see that D has an out-branching (i.e., $\ell_{\min}(D) > 0$) if and only if D has just one strongly connected component without incomming arcs [5]. Since the last condition can be checked in linear time [5], we may often assume that $\ell_{\min}(D) > 0$.

We first study MinLOB restricted to acyclic digraphs (abbreviated *MinLOB-DAG*). MinLOB-DAG was considered in US patent [10], where its application to the area of database systems was described. Demers and Downing [10] also suggested a heuristic approach to MinLOB-DAG. However no argument or assertion has been made to provide the validity of their approach and to investigate its computational complexity. Using another approach, we give a simple proof in Section 2 that MinLOB-DAG can be solved in polynomial time.

Since MinLOB generalizes the hamiltonian directed path problem, MinLOB is NP-hard. In this paper, we consider three parameterizations of MinLOB and show that two of them are NP-complete for every value of the parameter, but the third one is fixed-parameter tractable. The parameterized problems and related results are given in Sections 3 and 4. Further research is discussed in Section 5.

We recall some basic notions of parameterized complexity here, for a more in-depth treatment of the topic we refer the reader to [9,13,21].

A parameterized problem Π can be considered as a set of pairs (I, k) where I is the *problem instance* and k (usually an integer) is the *parameter*. Π is called *fixed-parameter tractable (FPT)* if membership of (I, k) in Π can be decided in time $O(f(k)|I|^c)$, where $|I|$ is the size of I, $f(k)$ is a computable function, and c is a constant independent from k and I. Let Π and Π' be parameterized problems with parameters k and k', respectively. An *fpt-reduction R from Π to Π'* is a many-to-one transformation from Π to Π', such that (i) $(I, k) \in \Pi$ if and only if $(I', k') \in \Pi'$ with $|I'| \leq g(k)$ for a fixed computable function g and (ii) R is of complexity $O(f(k)|I|^c)$. A *reduction to problem kernel* (or *kernelization*) is an fpt-reduction R from a parameterized problem Π to itself. In kernelization, an instance (I, k) is reduced to another instance (I', k'), which is called the *problem kernel*; $|I'|$ is the *size* of the kernel.

It is easy to see that a decidable parameterized problem is FPT if and only if it admits a kernelization (cf. [13,21]); however, the problem kernels obtained by this general result have impractically large size. Therefore, one tries to develop kernelizations that yield problem kernels of smaller size. The survey of Guo and Niedermeier [14] on kernelization lists some problem for which polynomial size kernels and exponential size kernels were obtained. Notice that if a kernelization can be done in time $O(n^{O(1)} + f(k))$, then we can obtain so-called an *additive FPT* algorithms, i.e., an algorithm of running time $O(n^{O(1)} + g(k))$, where $f(k)$ and $g(k)$ are independent of n, which is often significantly faster than its 'multiplicative' counterpart.

All digraphs in this paper are finite with no loops or parallel arcs. We use terminology and notation of [5]; in particular, for a digraph D, $V(D)$ and $A(D)$ denote its vertex and arc sets. The symbols n and m will denote the number of vertices and arcs in the digraph under consideration.

2 MinLOB-DAG

Let D be an acyclic digraph. We may assume that D has a unique vertex r of in-degree 0 as otherwise D has no out-branchings. Let $V = V(D)$ and $V' = \{v' : v \in V\}$. Let us define a bipartite graph B of D with partite sets X and X' as follows: $X = V$, $X' = V' \setminus \{r'\}$ and $E(B) = \{xy' : x \in X, y' \in X', xy \in A(D)\}$.

Consider the following algorithm for finding a minimum leaf out-branching T in an input acyclic digraph D. The algorithm outputs T if it exists and 'NO', otherwise.

MINLEAF

```
1. if the number of vertices with in-degree 0 equals 1 then
   $r \leftarrow$ the vertex of in-degree 0 else return 'NO'
2. construct the bipartite graph $B$ of $D$
3. find a maximum matching $M$ in $B$
4. $M^* \leftarrow M$
5. for all $y' \in X'$ not covered by $M$ do
   $M^* \leftarrow M^* \cup \{$an arbitrary edge incident with $y'\}$
6. $A(T) \leftarrow \emptyset$
7. for all $xy' \in M^*$ do $A(T) \leftarrow A(T) \cup \{xy\}$
8. return $T$
```

Theorem 1. *Let D be an acyclic digraph. Then MINLEAF returns a minimum leaf out-branching if one exists, or returns 'NO' otherwise in time $O(m + n^{1.5}\sqrt{m/\log n})$.*

Proof. We start with proving the validity of the algorithm. Observe that an acyclic digraph has an out-branching if and only if there exists only one vertex of in-degree zero. Hence Step 1 returns 'NO' precisely when $\ell_{\min}(D) = 0$.

Let M be the maximum matching obtained in Step 2, let $V(M)$ be the set of vertices of B covered by M, and let $Z = X \setminus V(M)$ and $Z' = X' \setminus V(M)$.

First we claim that Z is the set of the leaves of T, the out-branching of D obtained in the end of Step 7. Consider the edge set M^* obtained at the end of Step 5. First observe that for each vertex $y' \in Z'$, there exists an edge of $E(B)$ which is incident with y' since r is the only vertex of in-degree zero and thus no vertex of Z' is isolated. Moreover, all neighbors of y' are covered by M due to the maximality of M. It follows that $M^* \supseteq M$ covers all vertices of X' and leaves Z uncovered. Notice that r is covered by M. Indeed there exists a vertex u such that r is the only in-neighbor of u in D. Hence if r was not covered by M then u' would not be covered by M either, which means we could extend M by ru', a contradiction.

Consider T which has been obtained in the end of Step 7. Clearly $d_T^-(v) = 1$ for all $v \in V(D) \setminus \{r\}$ due to the construction of M^*. Moreover D does not have a cycle, which means that T is connected and thus is an out-branching. Finally no vertex of Z has an out-neighbor in T while all the other vertices have an out-neighbor. Now the claim holds.

Conversely, whenever there exists a minimum leaf out-branching T of D with the leaf set Z, we can build a matching in B which covers exactly $X \setminus Z$ among the vertices of X. Indeed, simply reverse the process of building an out-branching T from M^* described at Step 7. If some vertex $x \in X$ has more than one neighbor in X', eliminate all but one edge incident with x.

Secondly we claim that T obtained in MINLEAF(D) is of minimum number of leaves. Suppose to the contrary that the the attained out-branching T is not a minimum leaf out-branching of D. Then a minimum leaf out-branching can be used to produce a matching of B that covers more vertices of X than M

does using the argument in the preceding paragraph, a contradiction. Hence MINLEAF(D) returns a min leaf out-branching T at Step 8.

Finally we analyze the computational complexity of MINLEAF(D). Each step of MINLEAF(D) takes at most O(m) time except for Step 3. The computation time required to perform Step 3 is the same as that of solving the maximum cardinality matching problem on a bipartite graph. The last problem can be solved in time $O(|V(B)|^{1.5}\sqrt{|E(B)|/\log|V(B)|})$ [4]. Hence, the algorithm requires at most $O(m + n^{1.5}\sqrt{m/\log n})$ time. □

3 Parameterizations of MinLOB

The following is a natural way to parameterize MinLOB.

MinLOB Parameterized Naturally (MinLOB-PN)
Instance: A digraph D.
Parameter: A positive integer k.
Question: Is $\ell_{\min}(D) \leq k$?

Clearly, this problem is NP-complete already for $k = 1$ as for $k = 1$ MinLOB-PN is equivalent to the hamiltonian directed path problem. Let v be an arbitrary vertex of D. Transform D into a new digraph D_k by adding k vertices $v_1, v_2, \ldots, v_k$ together with the arcs $vv_1, vv_2, \ldots, vv_k$. Observe that D has a hamiltonian directed path terminating at v if and only if $\ell_{\min}(D) \leq k$. Since the problem is NP-complete of checking whether a digraph has a hamiltonian directed path terminating at a prescribed vertex, we conclude that MinLOB-PN is NP-complete for every fixed k.

Clearly, $\ell_{\min}(D) \leq n - 1$ for every digraph D of order n. Consider a different parameterizations of MinLOB.

MinLOB Parameterized Below Guaranteed Value (MinLOB-PBGV)
Instance: A digraph D of order n with $\ell_{\min}(D) > 0$.
Parameter: A positive integer k.
Question: Is $\ell_{\min}(D) \leq n - k$?
Solution: An out-branching B of D with at most $n - k$ leaves or the answer 'NO' to the above question.

Note that we consider MinLOB-PBGV as a search problem, not just as a decision problem. In the next section we will prove that MinLOB-PBGV is fixed-parameter tractable. We will find a problem kernel of order $O(k \cdot 2^k)$ and construct an additive FPT algorithm of running time $O(2^{O(k \log k)} + n^3)$. To obtain our results we use notions and properties of vertex cover and tree decomposition of underlying graphs and Las Vergnas' theorem on digraphs.

The parametrization MinLOB-PBGV is of the type *below a guaranteed value.* Parameterizations above/below a guaranteed value were first considered by Mahajan and Raman [20] for the problems Max-SAT and Max-Cut; such parameterizations have lately gained much attention, cf. [11,15,16,17,21] (it worth noting that Heggernes, Paul, Telle, and Villanger [17] recently solved the longstanding minimum interval completion problem, which is a parametrization above guaranteed value). For directed graphs there have been only a couple of results on problems parameterized above/below a guaranteed value, see [6,12].

Let us denote by $\boldsymbol{K}_{1,p-1}$ the *star digraph* of order p, i.e., the digraph with vertices $1, 2, \dots, p$ and arcs $12, 13, \dots, 1p$. Our success with MinLOB-PBGV may lead us to considering the following stronger (than MinLOB-PBGV) parameterizations of MinLOB.

MinLOB Parameterized Strongly Below Guaranteed Value (MinLOB-PSBGV)
Instance: A digraph D of order n with $\ell_{\min}(D) > 0$.
Parameter: An integer $k \geq 2$.
Question: Is $\ell_{\min}(D) \leq n/k$?

Unfortunately, MinLOB-PSBGV is NP-complete for every fixed $k \geq 2$. To prove this consider a digraph D of order n and a digraph H obtained from D by adding to it the star digraph $\boldsymbol{K}_{1,p-1}$ on $p = \lfloor n/(k-1) \rfloor$ vertices ($V(D) \cap V(\boldsymbol{K}_{1,p-1}) = \emptyset$) and appending an arc from vertex 2 of $\boldsymbol{K}_{1,p-1}$ to an arbitrary vertex y of D. Observe that $\ell_{\min}(H) = p - 1 + \ell_{min}(D, y)$, where $\ell_{min}(D, y)$ is the minimum possible number of leaves in an out-branching rooted at y, and that $\frac{1}{k}|V(H)| = p + \epsilon$, where $0 \leq \epsilon < 1$. Thus, $\ell_{\min}(H) \leq \frac{1}{k}|V(H)|$ if and only if $\ell_{min}(D, y) = 1$. Hence, the hamiltonian directed path problem with fixed initial vertex (vertex y in D) can be reduced to MinLOB-PSBGV for every fixed $k \geq 2$ and, therefore, MinLOB-PSBGV is NP-complete for every $k \geq 2$.

4 Solving MinLOB-PBGV

The *underlying graph* $UG(D)$ of a digraph D is obtained from D by omitting all orientation of arcs and by deleting one edge from each resulting pair of parallel edges. For a digraph D, let $\alpha(D)$ denote the independence number of $UG(D)$.

Theorem 2 (Las Vergnas[19]). *If a digraph D has an out-branching, then $\ell_{\min}(D) \leq \alpha(D)$.*

For an out-branching B of D, let $L(B)$ denote the set of leaves of B. We will prove the following claim which implies the theorem:

Claim 1. *Let B be an out-branching of D with more than $\alpha(D)$ leaves. Then D contains an out-branching B' such that $L(B')$ is a proper subset of $L(B)$.*

Proof. We will prove this claim by induction on the number n of vertices in D. For $n \leq 2$ the result holds; thus, we may assume that $n \geq 3$ and consider an out-branching B of D with $|L(B)| > \alpha(D)$. Clearly, D has an arc xy such that x, y are leaves of B. If the in-neighbor p of y in B is of out-degree at least 2, then $L(B') \subset L(B)$, where $B' = B + xy - py$. So, we may assume that $d^+_B(p) = 1$. Observe $\alpha(D - y) \leq \alpha(D) < |L(B)| = |L(B - y)|$. Hence by the induction hypothesis, $D-y$ has an out-branching B'' such that $L(B'') \subset L(B-y)$. Notice that $L(B-y) = L(B) \cup \{p\} \setminus \{y\}$. If $p \in L(B'')$, then observe that $L(B''+py) \subset L(B)$. Otherwise, $L(B''+xy) \subseteq L(B) \setminus \{x\} \subset L(B)$. □

A *vertex cover* of D is a vertex cover of $UG(D)$.

Lemma 1. *Let D be a digraph of order n with $\ell_{\min}(D) > 0$. In time $O(1.28^k + n^3)$, we can find either an out-branching of D with at most $n-k$ leaves or a vertex cover of D of size less than k.*

Proof. It is well-known that $\alpha(D) + \beta(D) = n$, where $\beta(D)$ is the minimum size of a vertex cover of D. First we can use the vertex cover algorithm of [8] to find a vertex cover of size less than k in time $O(1.2745^k k^4 + kn)$. If no such vertex cover exists, we have $\alpha(D) \leq n-k$ and by Theorem 2, D contains an out-branching B such that $|L(B)| \leq n-k$. To find such an out-branching we can use the procedure LEAFRED described below, which is just an algorithmic version of the proof of Claim 1.

The procedure LEAFRED(D, B) takes as an input a digraph D with $\beta(D) \geq k$ and an out-branching B of D and finds a new out-branching with leaves at most $\alpha(D)$. We may assume that the input out-branching B is a DFS tree. We denote by $p(y)$ the parent of a leaf vertex y in the given out-branching. The correctness of LEAFRED(D, B) follows from the proof of Claim 1 and it is not hard to see that its time complexity is $O(n^3)$.

```
**LEAFRED**$(D, B)$
improve $\leftarrow$ true
**while** improve=true **do**                                                    {
    **while** there is an arc $xy$ such that $x$ and $y$ are leaves and $d^+(p(y)) \geq 2$
        **do** $B \leftarrow B + xy - p(y)y$
    **if** there is an arc $xy$ such that $x$ and $y$ are leaves of $B$ **then** {
        $B'' \leftarrow$ LEAFRED$(D-y, B-y)$
        **if** $p(y) \in L(B'')$ **then** $B' \leftarrow B'' + p(y)y$
        **else** $B' \leftarrow B'' + xy$                                         }
    **else** $B' \leftarrow B$
    **if** $|L(B')| < |L(B)|$ **then** $B \leftarrow B'$ **else** improve $\leftarrow$ false          }
return $B$                                                                      □
```

It follows from Lemma 1 that there is an FPT algorithm that given an instance (D, k) of the MinLOB-PBGV problem either returns a solution or specifies a vertex cover of D of size less than k. We are going to show that, in the latter case, there is a possibility of kernelization. Let U be a vertex cover of D. Let $S \subseteq U$, $u \in U$ (we allow $S = \emptyset$). We denote by $V(S, u)$ a subset of vertices w of $V(D) \setminus U$ such that $(u, w) \in A(D)$ and $N^+(w) = S$. Let $NList$ be the set of all non-empty items $V(S, u)$. Now we create a set V' according to the following algorithm called **KERNEL**.

```
1. $V' \leftarrow \emptyset$
2. let all the elements of $NList$ be unmarked
3. **while** $NList$ has at least one unmarked item $V(S,u)$ **do**
      **if** $|V(S,u) \setminus V'| \leq 2|U|$ **then**
      $V' \leftarrow V' \cup V(S,u)$ **else**
      $V' \leftarrow V' \cup W$, where $W$ is an arbitrary subset of $2|U|$ vertices of $V(S,u) \setminus V'$
      **endif**
      mark $V(S,u)$
   **endwhile**
4. $V' \leftarrow V' \cup U$
5. **if** $D$ has a vertex $v^*$ with in-degree 0 **then** $V' \leftarrow V' \cup \{v^*\}$
6. return $V'$
```

Let D' be the subgraph of D induced by V'. The following lemma claims that D' can serve as a kernel of D with respect to the MinLOB-PBGV problem.

Lemma 2. *(D,k) is a 'YES' instance of the MinLOB-PBGV problem if and only if (D',k) is.*

Proof. In order to prove this lemma it is more convenient to think of the MinLOB-PBGV problem as a problem of constructing an out-branching with at least k non-leaf vertices rather than at most $n-k$ leaf vertices.

Assume that (D',k) is a 'YES' instance of the MinLOB-PBGV problem and let B' be an out-branching of D' having at least k non-leaf vertices. By definition of V', every vertex w of $V(D) \setminus V'$ has at least one in-neighbor $p(w)$ which belongs to $U \subseteq V'$. Add each such vertex w to B' together with arc $(p(w),w)$. Clearly, the resulting graph B is an out-branching of D whose set of non-leaf-vertices is a superset of the set of non-leaf vertices of B'. Thus, B has at least k non-leaf vertices which shows that (D,k) is a 'YES' instance of the MinLOB-PBGV problem.

Assume now that (D,k) is a 'YES' instance of the MinLOB-PBGV problem and let B be an out-branching of D having at least k non-leaf vertices. Let B^* be the subgraph of B induced by a set of vertices V^* defined as follows.

1. V^* contains all vertices of U and all the non-leaf vertices of B.
2. Let u be a non-leaf vertex such that the set $X(u)$ of children of u which are leaf vertices of B and belong to $V(D) \setminus U$ is non-empty. Then V^* contains exactly one vertex of $X(u)$.

Since all the non-leaf vertices of B belong to V^*, B^* is an out-tree. In addition, observe that every non-leaf vertex of B remains a non-leaf vertex in B^*. Indeed, consider an arbitrary non-leaf vertex u. If at least one child v of u is a non-leaf vertex itself or $v \in U$ then v is included in V^* according to the first item of definition of V^*. Otherwise, all the children of u are leaf vertices which belong to $V(D) \setminus U$. According to the second item of the definition of V^* at least one such child belongs to V^*. It follows that B^* is an out-tree with at least k non-leaf vertices.

Let $v_1, \ldots, v_t$ be the vertices of V^* enumerated in some arbitrary order. *Following this order*, we associate with each v_i a vertex $u_i \in V'$ according to the following procedure. If $v_i \in V'$ then $u_i = v_i$. Otherwise observe that the in-degree of v_i is at least one. Associate with v_i a vertex w such that w is the parent of v_i in B^* if v_i is a non-root and an arbitrary in-neighbor of v_i in D otherwise. Let u_i be an arbitrary vertex of $V(N^+(v_i), w) \cap V'$ which is not equal to u_j for any $j < i$.

Statement 1. *The procedure of construction of $u_1, \ldots, u_t$ is sound in the sense that if $v_i \notin V'$, the procedure always finds an available vertex for u_i.*

Proof of Statement 1. Observe that for $v_i \notin V'$, we have $|V(N^+(v_i), w) \cap V'| \geq 2|U|$ by the description of the kernelization algorithm. Since u_i is chosen among vertices in $V(N^+(v_i), w) \cap V'$ for each $v_i \notin V'$, it suffices to show that the number of vertices of $V(D) \setminus U$ among $u_1, \ldots, u_t$ does not exceed $2|U|$. Indeed, by construction of $u_1, \ldots, u_t$, we have $v_i \in U$ if and only if $u_i \in U$. Therefore, we may equivalently show that $|(V(D) \setminus U) \cap V^*| \leq 2|U|$.

The vertices of $V(D) \setminus U$ in V^* can be either non-leaf or leaf vertices. Each of the non-leaf vertices has a child in U and, of course, no two vertices share a child. Therefore the number of these vertices is at most U. On the other hand, by construction of V^*, each non-leaf vertex of B^* has at most one child which is a leaf vertex of $V(D) \setminus U$. Taking into account that the parent of a vertex of $V(D) \setminus U$ may be only a vertex of U, it follows that the number of vertices of the second category is also at most $|U|$ and the overall number of vertices of $V(D) \setminus U$ in B^* is at most $2|U|$ as required. □

Statement 2. *Let (v_i, v_j) be an arc of B^*. Then $(u_i, u_j) \in A(D')$.*

Proof of Statement 2. The statement is clearly true if $v_i = u_i$ and $v_j = u_j$. If $v_i \neq u_i$ but $v_j = u_j$, the statement follows because by selection of u_i, u_i has the same out-neighborhood as v_i. If $v_i = u_i$ but $v_j \neq u_j$ then the statement follows because by selection of u_j, the parent of v_j in B^* is an in-neighbor of u_j in D and hence in D'. Finally the case where $v_i \neq u_i$ and $v_j \neq u_j$ cannot happen because, by definition of a vertex cover, there is no arc between two vertices of $V(D) \setminus U$. □

It follows from the combination of statements that graph D' has a subgraph B_1 whose set of vertices is $u_1, \ldots, u_t$ and which is isomorphic to B^*. Observe that any vertex of $V' \setminus V(B_1)$ has an in-neighbor among the vertices of B_1. Indeed, any vertex w of $V' \setminus V(B_1)$ belongs to $V(D) \setminus U$. Hence all the in-neighbors of w in D belong to U and thus to $V(B_1)$. Consequently, if w does not have in-neighbors in $V(B_1)$, the in-degree of w in D is 0. It follows that w is the root vertex of B and hence the root vertex of B^*. By construction of $u_1, \ldots, u_t$, we have w is necessarily one of u_i-s, a contradiction. Therefore for each vertex w of $V' \setminus V(B_1)$, we can select an in-neighbor $p(w) \in V(B_1)$ of w in D'. Add vertex w and arc $(p(w), w)$ to B_1 for each $w \in V' \setminus V(B_1)$, and let B_2 be the resulting digraph. It is not hard to see that B_2 is an out-branching of D' with at least k non-leaf vertices, which shows that (D', k) is a 'YES' instance of the MinLOB-PBGV problem. □

Combining Lemmas 1 and 2 we obtain the following theorem.

Theorem 3. *The MinLOB-PBGV problem is FPT. In particular, there is an $O(1.28^k + n^3)$ time algorithm which given an instance (D, k) of the MinLOB-PBGV problem, either produces a solution or reduces the instance (D, k) to an instance (D', k) where $|V(D')| = O(k \cdot 2^k)$.*

Proof. It follows from Lemma 1 that there is an $O(1.28^k + n^3)$ algorithm which finds either vertex cover U of D of size at most $k - 1$ or produces an out-branching with at most $n - k$ leaves. Assume that the algorithm has found a vertex cover U such that $|U| < k$. Consider the transformation from (D, k) to (D', k) performed by the algorithm KERNEL *given* the vertex cover U. Lemma 2 proves that (D, k) is a 'YES' instance if and only if (D', k) is. We will show that this transformation takes $O(n^3)$ time and $|V(D')| = O(k \cdot 2^k)$ and that will complete the proof.

For $S \subseteq U$, let $V_S = \{x \in V(D) \setminus U : N^+(x) = S\}$. Going through all vertices of D one by one and comparing their out-neighborhoods, we can find all sets V_S in time $O(n^3)$. Now for each fixed $u \in U$ we can construct all non-empty sets $V(S, u)$ in time $O(n^2)$. Thus, all non-empty sets $V(S, u)$ can be found in time $O(n^3)$. Given the whole list $NList$, the construction of V' requires $O(n^3)$.

Let us now compute the number of vertices of D'. In terms of $|U|$, the number of elements of sets $V(S, u)$ which belong to V' is at most $2|U| \cdot 2^{|U|}$. In addition at most $|U| + 1$ vertices are added to V' at the end of KERNEL. Thus the number of vertices of V' is $O(|U| \cdot 2^{|U|}) = O(k \cdot 2^k)$ since $|U| < k$. □

Thus we have shown that the MinLOB-PBGV problem has a kernel of order proportional to $k \cdot 2^k$. Now we are going to clarify how we explore this kernel in order to get the desired out-branching. A straightforward exploration of all possible out-branchings (using, e.g., the main algorithm of [18]) is not a good choice because the number of different out-branchings may be up to p^{p-1}, where $p = |V(D')| = (k \cdot 2^k)$. Indeed, by the famous Kelly's formula the number of spanning trees in the complete graph K_p on p vertices equals p^{p-2}. In the complete digraph on p vertices, one can get p out-branchings from each spanning tree of K_p by assigning a vertex to be the root.

In order to achieve a better running time we provide an alternative way of showing the fixed-parameter tractability of the MinLOB-PBGV problem based on the notion of *tree decomposition*.

A *tree decomposition* of an (undirected) graph G is a pair (X, U) where U is a tree whose vertices we will call *nodes* and $X = \{X_i : i \in V(U)\}$ is a collection of subsets of $V(G)$ (called *bags*) such that

1. $\bigcup_{i \in V(U)} X_i = V(G)$,
2. for each edge $\{v, w\} \in E(G)$, there is an $i \in V(U)$ such that $v, w \in X_i$, and
3. for each $v \in V(G)$ the set of nodes $\{i : v \in X_i\}$ form a subtree of U.

The *width* of a tree decomposition $(\{X_i : i \in V(U)\}, U)$ equals $\max_{i \in V(U)}\{|X_i| - 1\}$. The *treewidth* of a graph G is the minimum width over all tree decompositions of G. We use the notation $\mathrm{tw}(G)$ to denote the treewidth of a graph G.

By a *tree decomposition of a digraph D* we will mean a tree decomposition of the underlying graph $UG(D)$. Also, $\mathrm{tw}(D) = \mathrm{tw}(UG(D))$.

Theorem 4. *There is a polynomial algorithm that, given an instance (D, k) of the MinLOB-PBGV problem, either finds a solution or establishes a tree decomposition of D of width at most k.*

Proof. By Lemma 1, there is a polynomial algorithm which either finds a solution or specifies a vertex cover C of D of size at most k. Let $I = \{v_1, \dots, v_s\} = V(D) \setminus C$. Consider a star U with nodes $x_0, x_1, \dots, x_s$ and edges $x_0x_1, x_0x_2, \dots, x_0x_s$. Let $X_0 = C$ and $X_i = X_0 \cup \{v_i\}$ for $i = 1, 2, \dots, s$ and let X_j be the bag corresponding to x_j for every $j = 0, 1, \dots, s$. Observe that $(\{X_0, X_1, \dots, X_s\}, U)$ is a tree decomposition of D and its width is at most k. □

Theorem 4 shows that an instance (D, k) of the MinLOB-PBGV problem can be reduced to another instance with treewidth $O(k)$. Using standard dynamic programming techniques we can solve this instance in time $2^{O(k \log k)} n^{O(1)}$. On the first glance it seems that this running time makes the above kernelization redundant. However, although the $O(k \cdot 2^k)$ kernel is not polynomial, it is much smaller than $2^{O(k \log k)}$. Therefore if we first find the kernel and then establish the tree decomposition, the resulting dynamic programming algorithm will run in time $2^{O(k \log k)} + n^{O(1)}$ *without* changing the constant at $k \log k$. More precisely, Theorem 3 and Theorem 4 imply the following corollary.

Corollary 1. *Let D a digraph of order n. Suppose that a tree-decomposition of D of width k is specified. Suppose also that given this tree-decomposition the MinLOB-PBGV problem can be solved in time $2^{ck \log k} n^{O(1)}$. Then for any instance (D, k), the MinLOB-PBGV problem can be solved in time $O(2^{ck \log k + dk} + 1.28^k + n^3)$, where d is a constant.*

Proof. The additional dk at the exponent follows from replacing $n^{O(1)}$ by $(k2^k)^{O(1)}$. To obtain a vertex cover of size at most k, if one exists, takes $O(1.28^k + kn)$ time. The kernelization can be done in $O(n^3)$ time by Theorem 3. □

The above results imply the following:

Theorem 5. *The MinLOB-PBGV problem can be solved by an additive FPT algorithm of running time $O(2^{O(k \log k)} + n^3)$.*

5 Discussion and Further Research

We have proved that MinLOB-PBGV is FPT. It would be interesting to check whether MinLOB-PBGV admits significantly more efficient FPT algorithms, i.e., algorithms of complexity $O(c^k n^{O(1)})$, where c is a constant. The same question is of interest for the following related problem, which is the natural parametrization of the Maximum Leaf Out-Branching problem.

MaxLOB Parameterized Naturally (MaxLOB-PN)
Instance: A digraph D.
Parameter: A positive integer k.
Question: Does D have an out-branching with at least k leaves?

Alon et al. [1,2] proved that this problem is FPT for several special classes of digraphs such as strongly connected digraphs and acyclic digraphs, and Bonsma and Dorn [7] proved that the problem is FPT. Note that in the three papers, MaxLOB-PN algorithms are of running time $O(2^{k(\log k)^{O(1)}} \cdot n^{O(1)})$.

Interestingly, MaxLOB-PN remains NP-complete even when the given digraph D is acyclic [3], which is in a clear contrast with MinLOB-PBGV unless P=NP.

Acknowledgements. Research of Gutin and Kim was supported in part by an EPSRC grant. Part of the paper was written when Razgon was vising Department of Computer Science, Royal Holloway, University of London. The research of Razgon at the Department of Computer Science, University College Cork was supported by Science Foundation Ireland Grant 05/IN/I886.

References

1. Alon, N., Fomin, F., Gutin, G., Krivelevich, M., Saurabh, S.: Parameterized Algorithms for Directed Maximum Leaf Problems. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 352–362. Springer, Heidelberg (2007)
2. Alon, N., Fomin, F., Gutin, G., Krivelevich, M., Saurabh, S.: Better Algorithms and Bounds for Directed Maximum Leaf Problems. In: Arvind, V., Prasad, S. (eds.) FSTTCS 2007. LNCS, vol. 4855, Springer, Heidelberg (2007)
3. Alon, N., Fomin, F.V., Gutin, G., Krivelevich, M., Saurabh, S.: Spanning directed trees with many leaves. Report arXiv: 0803.0701 (2008)
4. Alt, H., Blum, N., Melhorn, K., Paul, M.: Computing of maximum cardinality matching in a bipartite graph in time $O(n^{1.5}\sqrt{m/\log n})$. Inf. Proc. Letters 37, 237–240 (1991)
5. Bang-Jensen, J., Gutin, G.: Digraphs: Theory, Algorithms and Applications. Springer, Heidelberg (2000), `www.cs.rhul.ac.uk/books/dbook/`
6. Bang-Jensen, J., Yeo, A.: The minimum spanning strong subdigraph problem is fixed parameter tractable. Discrete Applied Math. (to appear)
7. Bonsma, P.S., Dorn, F.: An FPT Algorithm for Directed Spanning k-Leaf. Preprint 046-2007, Combinatorial Optimization & Graph Algorithms Group, TU Berlin (November 2007) (preprint 046-2007)
8. Chandran, L.S., Grandoni, F.: Refined memorization for vertex cover. Inform. Proc. Letters 93, 125–131 (2005)
9. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, Heidelberg (1999)

10. Demers, A., Downing, A.: Minimum leaf spanning tree. US Patent no. 6,105,018 (August 2000)
11. Fernau, H.: Parameterized Algorithmics: A Graph-theoretic Approach. Habilitation thesis, U. Tübingen (2005)
12. Fernau, H.: Parameterized Algorithmics for Linear Arrangement Problems(manscript, July 2005)
13. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Heidelberg (2006)
14. Guo, J., Niedermeier, R.: Invitation to Data Reduction and Problem Kernelization. ACM SIGACT News 38, 31–45 (2007)
15. Gutin, G., Rafiey, A., Szeider, S., Yeo, A.: The Linear Arrangement Problem Parameterized Above Guaranteed Value. Theory of Computing Systems 41, 521–538 (2007)
16. Gutin, G., Szeider, S., Yeo, A.: Fixed-Parameter Complexity of Minimum Profile Problems. Algorithmica (to appear)
17. Heggernes, P., Paul, C., Telle, J.A., Villanger, Y.: Interval completion with few edges. In: Proc. STOC 2007 - 39th ACM Symposium on Theory of Computing, pp. 374–381 (2007)
18. Kapoor, S., Ramesh, H.: An Algorithm for Enumerating All Spanning Trees of a Directed Graph. Algorithmica 27, 120–130 (2000)
19. Las Vergnas, M.: Sur les arborescences dans un graphe orienté. Discrete Math. 15, 27–29 (1976)
20. Mahajan, M., Raman, V.: Parameterizing above guaranteed values: MaxSat and MaxCut. J. Algorithms 31, 335–354 (1999)
21. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press, Oxford (2006)