Parameterized Complexity of DAG Partitioning

René van Bevern1,*, Robert Bredereck1,⋆, Morgan Chopin2,⋆⋆, Sepp Hartung1, Falk Hüffner1,†, André Nichterlein1, and Ondřej Suchý3,‡

1 Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany
{rene.vanbevern,robert.bredereck,sepp.hartung,
falk.hueffner,andre.nichterlein}@tu-berlin.de
2 LAMSADE, Université Paris-Dauphine, France
chopin@lamsade.dauphine.fr
3 Faculty of Information Technology, Czech Technical University in Prague,
Prague, Czech Republic
ondrej.suchy@fit.cvut.cz

Abstract. The goal of tracking the origin of short, distinctive phrases (memes) that propagate through the web in reaction to current events has been formalized as DAG PARTITIONING: given a directed acyclic graph, delete edges of minimum weight such that each resulting connected component of the underlying undirected graph contains only one sink. Motivated by NP-hardness and hardness of approximation results, we consider the parameterized complexity of this problem. We show that it can be solved in $O(2^k \cdot n^2)$ time, where $k$ is the number of edge deletions, proving fixed-parameter tractability for parameter $k$. We then show that unless the Exponential Time Hypothesis (ETH) fails, this cannot be improved to $2^{o(k)} \cdot n^{O(1)}$; further, DAG PARTITIONING does not have a polynomial kernel unless NP ⊆ coNP/poly. Finally, given a tree decomposition of width $w$, we show how to solve DAG PARTITIONING in $2^{O(w^2)} \cdot n$ time, improving a known algorithm for the parameter pathwidth.

1 Introduction

The motivation of our problem comes from a data mining application. Leskovec et al. [6] want to track how short phrases (typically, parts of quotations) show up on different news sites, sometimes in mutated form. For this, they collected from 90 million articles phrases of at least four words that occur at least ten times. They then created a directed graph with the phrases as vertices and draw an arc from phrase $p$ to phrase $q$ if $p$ is shorter than $q$ and either $p$ has small edit distance from $q$ (with words as tokens) or there is an overlap of at least 10 consecutive words. Thus, an arc $(p, q)$ indicates that $p$ might originate from $q$.  

* Supported by DFG project DAPA (NI 369/12-1).
** Supported by DFG project PAWS (NI 369/10-1).
*** Supported by DAAD.
† Supported by DFG project PABI (NI 369/7-2).
‡ A major part of this work was done while with the TU Berlin, supported by the DFG project AREG (NI 369/9).
Since all arcs are from shorter to longer phrases, the graph is a directed acyclic graph (DAG). The arcs are weighted according to the edit distance and the frequency of \( q \). A vertex with no outgoing arc is called a sink. If a phrase is connected to more than one sink, its ultimate origin is ambiguous. To resolve this, Leskovec et al. introduce the following problem.

**DAG Partitioning**

Input: A directed acyclic graph \( D = (V, A) \) with positive integer edge weights \( \omega : A \rightarrow \mathbb{N} \) and a positive integer \( k \in \mathbb{N} \).

Output: Is there a set \( A' \subseteq A \), \( \sum_{a \in A'} \omega(a) \leq k \), such that each connected component in \( D' = (V, A \setminus A') \) has exactly one sink?

While the work of Leskovec et al. had a large impact (for example, it was featured in the New York Times), there are few studies on the computational complexity of DAG Partitioning so far. Leskovec et al. show that DAG Partitioning is NP-hard. Alamdari and Mehrabian show that moreover it is hard to approximate in the sense that if \( P \neq NP \), then for any fixed \( \varepsilon > 0 \), there is no \( (n^{1-\varepsilon}) \)-approximation, even if the input graph is restricted to have unit weight arcs, maximum outdegree three, and two sinks.

In this paper, we consider the parameterized complexity of DAG Partitioning. (We assume familiarity with parameterized analysis and concepts such as problem kernels (see e.g. [4, 7])). Probably the most natural parameter is the maximum weight \( k \) of the deleted edges; edges get deleted to correct errors and ambiguity, and we can expect that for sensible inputs only few edges need to be deleted.

Unweighted DAG Partitioning is similar to the well-known Multiway Cut problem: given an undirected graph and a subset of the vertices called the terminals, delete a minimum number \( k \) of edges such that each terminal is separated from all others. DAG Partitioning in a connected graph can be considered as a Multiway Cut problem with the sinks as terminals and the additional constraint that not all edges going out from a vertex may be deleted, since this creates a new sink. Xiao gives a fixed-parameter algorithm for solving Multiway Cut in \( O(2^k \cdot n^{O(1)}) \) time. We show that a simple branching algorithm solves DAG Partitioning in the same running time (Theorem 3). We also give a matching lower bound: unless the Exponential Time Hypothesis (ETH) fails, DAG Partitioning cannot be solved in \( O(2^{o(k)} \cdot n^{O(1)}) \) time (Corollary 1). We then give another lower bound for this parameter by showing that DAG Partitioning does not have a polynomial kernel unless \( NP \subseteq \text{coNP/poly} \) (Theorem 5).

An alternative parameterization considers the structure of the underlying undirected graph of the input. Alamdari and Mehrabian show that if this graph has pathwidth \( \phi \), DAG Partitioning can be solved in \( 2^{O(\phi^2)} \cdot n \) time, and thus DAG Partitioning is fixed-parameter tractable with respect to pathwidth. They ask if DAG Partitioning is also fixed-parameter tractable with respect to the parameter treewidth. We answer this question positively by giving an algorithm based on dynamic programming that given a tree decomposition of width \( w \) solves DAG Partitioning in \( O(2^{O(w^2)} \cdot n) \) time (Theorem 7).

Due to space constraints, we defer some proofs to a journal version.