View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Dagstuhl Research Online Publication Server

A Parameterized Algorithmics Framework for Degree Sequence Completion Problems in Directed Graphs*

Robert Bredereck[1], Vincent Froese[2], Marcel Koseler[3], Marcelo Garlet Millani[4], André Nichterlein†[5], and Rolf Niedermeier[6]

1 **Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany**
`robert.bredereck@tu-berlin.de`
2 **Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany**
`vincent.froese@tu-berlin.de`
3 **Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany**
`marcel.koseler@campus.tu-berlin.de`
4 **Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany**
`marcelo.garletmillani@campus.tu-berlin.de`
5 **Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany**
`andre.nichterlein@tu-berlin.de`
6 **Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany**
`rolf.niedermeier@tu-berlin.de`

Abstract

There has been intensive work on the parameterized complexity of the typically NP-hard task to edit undirected graphs into graphs fulfilling certain given vertex degree constraints. In this work, we lift the investigations to the case of directed graphs; herein, we focus on arc insertions. To this end, our general two-stage framework consists of efficiently solving a problem-specific number problem transferring its solution to a solution for the graph problem by applying flow computations. In this way, we obtain fixed-parameter tractability and polynomial kernelizability results, with the central parameter being the maximum vertex in- or outdegree of the output digraph. Although there are certain similarities with the much better studied undirected case, the flow computation used in the directed case seems not to work for the undirected case while f-factor computations as used in the undirected case seem not to work for the directed case.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.2.2 Graph Theory

Keywords and phrases NP-hard graph problem, graph realizability, graph modification, arc insertion, fixed-parameter tractability, kernelization

Digital Object Identifier 10.4230/LIPIcs.IPEC.2016.10

1 Introduction

Modeling real-world networks (e.g., communication, ecological, social) often requests *directed* graphs (digraphs for short). We study a class of specific "network design" (in the sense of

* A full version of the paper is available at `https://arxiv.org/abs/1604.06302`.
† From Feb. 2016 to Jan. 2017 on postdoctoral leave to Durham University (GB), funded by DAAD.

© Robert Bredereck, Vincent Froese, Marcel Koseler, Marcelo Garlet Millani, André Nichterlein, and Rolf Niedermeier;
licensed under Creative Commons License CC-BY

constructing a specific network topology) or "graph realization" problems. Here, our focus is on inserting arcs into a given digraph in order to fulfill certain vertex degree constraints. These problems are typically NP-hard, so we choose parameterized algorithm design for identifying relevant tractable special cases. The main parameter we work with is the maximum in- or outdegree of the newly constructed digraph. To motivate the problems we deal with, consider the following three application scenarios.

1. Assume we are given a directed network representing a system's current state. Then, each individual node might have certain desired states of connectivity in terms of the numbers of in- and outgoing arcs which we want to satisfy by inserting arcs between the nodes. For instance, in a peer-review network we have an arc from one author reviewing a paper of another author. Depending on research experience, the authors might have different requests with respect to the number of own papers to be reviewed by others and other papers which they are reviewing. This leads to the Digraph Degree Constraint Completion problem as studied in Section 4.1.
2. Assume that we have two different data sources: A network which is an incomplete measurement of some unreliable source and the true degree sequence of the target network. The goal is to reconstruct the original network by inserting arcs such that we obtain the target degree sequence (in a sense, the network matches the given degree sequence). In the presence of labeled input networks this might for example reveal communication patterns between users in social networks. The corresponding problem is called Digraph Degree Sequence Completion and studied in Section 4.2.
3. Assume we want to "k-anonymize" a social network, that is, after inserting a minimum number of arcs each degree, that is, each combination of in- and outdegree, occurs either zero or at least k times. This leads to the Digraph Degree Anonymity problem as studied in Section 4.3.

All three problems are NP-hard. Based on a general framework presented in Section 3, we derive several fixed-parameter tractability results for them, mainly exploiting the parameter "maximum vertex degree" in the output digraph. Moreover, the three problems above are special cases of the Digraph Degree Constraint Sequence Completion problem which we will define next.

These three problems as well as Digraph Degree Constraint Sequence Completion are special graph modification problems: given a graph, can it be changed by a minimum number of graph modifications such that the resulting graph adheres to specific constraints for its degree sequence?

In the most basic variant a degree sequence is a sequence of positive integers specifying (requested) vertex degrees for a fixed ordering of the vertices. Typically, the corresponding computational problems are NP-hard. In recent years, research in this direction focused on undirected graphs [8, 10, 11, 14, 18, 20]. In this work, we investigate parameterized algorithms on digraphs. As Gutin and Yeo [12] observed, much less is known about the structure of digraphs than that of undirected graphs making the design of parameterized algorithms for digraphs more challenging. In particular, we present a general framework for a class of degree sequence modification problems, focusing on the case of arc insertions (that is, completion problems).

The most general degree completion problem for digraphs we consider in this work is as follows.

DIGRAPH DEGREE CONSTRAINT SEQUENCE COMPLETION (DDCONSEQC)
Input: A digraph $D = (V, A)$, a non-negative integer s, a "degree list function" $\tau\colon V \to 2^{\{0,\ldots,r\}^2}$, and a "sequence property" Π.
Question: Is it possible to obtain a digraph D' by inserting at most s arcs in D such that the degree sequence of D' fulfills Π and $\deg_{D'}(v) \in \tau(v)$ for all $v \in V$?

We emphasize that there are two types of constraints – one (specified by the function τ) for the individual vertices and one (specified by Π) for the whole list of degree tuples. For instance, a common Π as occurring in the context of data privacy applications is to request that the list is k*-anonymous*, that is, every combination of in- and outdegree that occurs in the list occurs at least k times (see the third motivating example before).

Since DDCONSEQC and its special cases as studied here all turn out to be NP-hard [19, 15], a parameterized complexity analysis seems the most natural fit for understanding the computational complexity landscape of these kinds of problems – this has also been observed in the above mentioned studies for the undirected case. Our main findings are mostly on the positive side. That is, although seemingly more intricate to deal with due to the existence of in- and outdegrees, many positive algorithmic results which hold for undirected graphs can also be achieved for digraphs (albeit using different techniques). In particular, we present a maximum-flow-based framework that, together with the identification and solution of certain number problems, helps to derive several fixed-parameter tractability results with respect to the parameter maximum possible in- or outdegree Δ^* in any solution digraph. Notably, the corresponding result in the undirected case was based on f-factor computations [8] which do not transfer to the directed case, and, vice versa, the flow computation approach we present for the directed case seemingly does not transfer to the undirected case. For special cases of DDCONSEQC, we can move further and even derive some polynomial-size problem kernels, again for the parameter Δ^*.

We consider the parameter Δ^* for the following reasons. First, it is always at most r, a natural parameter in the input. Second, in combination with Π, we might get an even smaller upper bound for Δ^*. Third, bounded-degree graphs are well studied and our work extends this since we only require Δ^* to be small, not to be constant.

Related Work. Most of the work on graph modification problems for realizing degree constraints has focused on undirected graphs [8, 10, 11, 14, 18, 20]. Closest to our work is the framework for deriving polynomial-size problem kernels for undirected degree sequence completion problems [8], which we complement by our results for digraphs. Generally, we can derive similar results, but the technical details differ and the landscape of problems is richer in the directed case. As to digraph modification problems in general, we are aware of surprisingly little work. We mention work studying arc insertion for making a digraph transitive [22] or for making a graph Eulerian [7], both employing the toolbox of parameterized complexity analysis. Somewhat related is also work about the insertion of edges into a mixed graph to satisfy local edge-connectivity constraints [1] or about orienting edges in a partially oriented graph to make it an oriented graph [2].

Our Results. In Section 3, we present our general framework for DDCONSEQC. That is, based on flow computations, in a two-stage approach we show that it is fixed-parameter tractable with respect to the parameter Δ^*. To this end, we identify a specific pure number problem that needs to be fixed-parameter tractable with respect to the largest integer in the input. Next, presenting applications of the framework, in Section 4.1, we show that if there is no constraint Π concerning the degree sequence (that is, DIGRAPH DEGREE CONSTRAINT

Completion), then we not only obtain fixed-parameter tractability but also a polynomial-size problem kernel for parameter Δ^* can be obtained. Then, in Section 4.2 we show an analogous result if there is one exactly specified degree sequence to be realized (Digraph Degree Sequence Completion). Finally, in Section 4.3, we show that if we request the degree sequence to be k-anonymous (that is, Digraph Degree Anonymity), then we can at least derive a polynomial-size problem kernel for the combined parameter (s, Δ_D), where Δ_D denotes the maximum in- or outdegree of the input digraph D. Also, we take a first step outlining the limitations of our framework for digraphs. In contrast to the undirected case (which is polynomial-time solvable [17]), the corresponding number problem of Digraph Degree Anonymity surprisingly is weakly NP-hard and presumably not polynomial-time solvable. Due to lack of space, several proofs are deferred to a full version (available at `https://arxiv.org/abs/1604.06302`).

2 Preliminaries

We consider *digraphs* (without multiarcs or loops) $D = (V, A)$ with $n := |V|$ and $m := |A|$. For a vertex $v \in V$, $\deg^-_D(v)$ denotes the *indegree* of v, that is, the number of incoming arcs of v. Correspondingly, $\deg^+_D(v)$ denotes the *outdegree*, that is, the number of outgoing arcs of v. We define the *degree* $\deg_D(v) := (\deg^-_D(v), \deg^+_D(v))$. The set $V(A') := \{v \in V \mid ((v,w) \in A' \vee (w,v) \in A') \wedge w \in V\}$ contains all vertices incident to an arc in $A' \subseteq V^2$. For a set of arcs $A' \subseteq V^2$, $D + A'$ denotes the digraph $(V, A \cup A')$, while $D[A']$ denotes the subdigraph $(V(A'), A')$. Analogously, for a set of vertices $V' \subseteq V$, $D[V']$ denotes the induced subdigraph $(V', A \cap (V')^2)$ which only contains the vertices V' and the arcs between vertices from V'. The set $N^+_D(v) := \{w \in V \mid (v,w) \in A\}$ denotes the set of *outneighbors* of v. Analogously, $N^-_D(v) := \{w \in V \mid (w,v) \in A\}$ denotes the set of *inneighbors*. Furthermore, we define the maximum indegree $\Delta^-_D := \max_{v \in V} \deg^-_D(v)$, the maximum outdegree $\Delta^+_D := \max_{v \in V} \deg^+_D(v)$, and $\Delta_D := \max\{\Delta^+_D, \Delta^-_D\}$.

A *digraph degree sequence* $\sigma = \{(d^-_1, d^+_1), \dots, (d^-_n, d^+_n)\}$ is a multiset of nonnegative integer tuples, where $d^-_i, d^+_i \in \{0, \dots, n-1\}$ for all $i \in \{1, \dots, n\}$. We define

$$\Delta^-_\sigma := \max\{d^-_1, \dots, d^-_n\}, \qquad \Delta^+_\sigma := \max\{d^+_1, \dots, d^+_n\}, \text{ and } \qquad \Delta_\sigma := \max\{\Delta^-_\sigma, \Delta^+_\sigma\}.$$

For a digraph $D = (\{v_1, \dots, v_n\}, A)$ we denote by $\sigma(D) := \{\deg_D(v_1), \dots, \deg_D(v_n)\}$, the digraph degree sequence of D. Let $d = (d^-, d^+)$ be a nonnegative integer tuple. For a digraph D, the *block* $B_D(d)$ *of degree* d is the set of all vertices having degree d, formally $B_D(d) := \{v \in V \mid \deg_D(v) = d\}$. We define $\lambda_D(d)$ as the number of vertices in D with degree d, that is, $\lambda_D(d) := |B_D(d)|$. Similarly, we define $B_\sigma(t)$ as the multiset of all tuples equal to t and $\lambda_\sigma(t)$ as the number of occurrences of the tuple t in the multiset σ. For two integer tuples (x_1, y_1), (x_2, y_2), we define the sum $(x_1, y_1) + (x_2, y_2) := (x_1 + x_2, y_1 + y_2)$.

3 The Framework

Our goal is to develop a framework for deriving fixed-parameter tractability for a general class of completion problems in directed graphs. To this end, recall our general setting for DDConSeqC which is as follows. We are given a digraph and want to insert at most s arcs such that the vertices satisfy certain degree constraints τ, and additionally, the degree sequence of the digraph fulfills a certain property Π. Formally, the sequence property Π is given as a function that maps a digraph degree sequence to 1 if the sequence fulfills the property and otherwise to 0. We restrict ourselves to properties where the corresponding function can be encoded with only polynomially many bits in the number of vertices of the input digraph and can be decided efficiently.[1] We remark that it is not always the case that there are both vertex degree constraints (as defined by τ) and degree sequence constraints (as defined by Π) requested. This can be handled by either setting τ to the trivial degree list function with $\tau(v) = \{0, \ldots, n-1\}^2$ for all $v \in V$ or setting Π to allow all possible degree sequences.

In this section, we show how to derive (under certain conditions) fixed-parameter tractability with respect to the maximum possible in- or outdegree Δ^* of the *output* digraph for DDConSeqC. Note that Δ^* in general is not known in advance. In practice, we might therefore instead consider upper bounds for Δ^* which depend on the given input. For example, it always holds $\Delta^* \leq \min\{r, \Delta_D + s\}$ since we are only inserting at most s arcs in D. Clearly, Δ^* might also be upper-bounded depending on Π (or even depending on r, s, Δ_D, and Π) in some cases. Our generic framework consists of two main steps: First, we prove fixed-parameter tractability with respect to the combined parameter (s, Δ_D) in Section 3.1. This step generalizes ideas for the undirected case [8]. Note that $\Delta_D \leq \Delta^*$ trivially holds. Second, we show in Section 3.2 how to upper-bound the number s of arc insertions polynomially in Δ^* by solving a certain problem specific numerical problem. For this step, we develop a new key argument based on a maximum flow computation (the undirected case was based on f-factor arguments).

3.1 Fixed-parameter tractability with respect to (s, Δ_D)

We show that DDConSeqC is fixed-parameter tractable with respect to the combination of the maximum number s of arcs to insert and the maximum in- or outdegree Δ_D of the input digraph D. The basic idea underlying this result is that two vertices v and w with $\deg_D(v) = \deg_D(w)$ and $\tau(v) = \tau(w)$ are interchangeable. Accordingly, we will show that it suffices to consider only a bounded number of vertices with the same "degree properties". In particular, if there is a solution, then there is also a solution that only inserts arcs between a properly chosen subset of vertices of bounded size. To formalize this idea, we introduce the notion of an *α-block-type set* for some positive integer α.

To start with, we define the types of a vertex via the numbers of arcs that τ allows to add to this vertex. Let (D, s, τ, Π) be a DDConSeqC instance. A vertex v is of *type* $t \in \{0, \ldots, \Delta^*\}^2$ if $\deg_D(v) + t \in \tau(v)$. Observe that one vertex can be of several types. The subset of $V(D)$ containing all vertices of type t is denoted by $T_{D,\tau}(t)$. A vertex v of type $(0,0)$ (that is, $\deg_D(v) \in \tau(v)$) is called *satisfied*. A vertex which is not satisfied is called *unsatisfied*. We next define our notion of α-block-type sets and its variants.

▶ **Definition 1.** Let α be a positive integer and let $U \subseteq V(D)$ denote the set of all unsatisfied vertices in D. A vertex subset $C \subseteq V(D)$ with $U \subseteq C$ is called

- *α-type set* if, for each type $t \neq (0,0)$, C contains exactly $\min\{|T_{D,\tau}(t) \setminus U|, \alpha\}$ satisfied vertices of type t;
- *α-block set* if, for each degree $d \in \sigma(D)$, C contains exactly $\min\{|B_D(d) \setminus U|, \alpha\}$ satisfied vertices with degree d;
- *α-block-type set* if, for each degree $d \in \sigma(D)$ and each type $t \neq (0,0)$, C contains exactly $\min\{|(B_D(d) \cap T_{D,\tau}(t)) \setminus U|, \alpha\}$ satisfied vertices of degree d and type t.

[1] All specific properties in this work can be easily decided in polynomial time. Indeed, in many cases even fixed-parameter tractability with respect to the maximum integer in the sequence would suffice.

As a first step, we prove that these sets defined above can be computed efficiently.

▶ **Lemma 2.** *An α-type/α-block/α-block-type set C as described in Definition 1 can be computed in $O(m + |\tau| + r^2)$ / $O(m + n + \Delta_D^2)$ / $O(m + |\tau| + \Delta_D^2 r^2)$ time.*

We move on to the crucial lemma stating that a solution (that is, a set of arcs), if existing, can always be found in between vertices of an α-block-type set C given that C contains "enough" vertices of each degree and type. Here, enough means $\alpha := 2s(\Delta_D + 1)$.

▶ **Lemma 3.** *Let (D, s, τ, Π) be a* DDConSeqC *instance and let $C \subseteq V(D)$ be a $2s(\Delta_D + 1)$-block-type set. If (D, s, τ, Π) is a yes-instance, then there exists a solution $A^* \subseteq C^2$ for (D, s, τ, Π), that is, $|A^*| \leq s$, $\sigma(D + A^*)$ fulfills Π, and $\deg_{D+A^*}(v) \in \tau(v)$ for all $v \in V(D)$.*

If there are no restrictions on the resulting degree sequence (as it is the case for the Digraph Degree Constraint Completion problem (DDConC) in Section 4.1), then we can replace the $2s(\Delta_D + 1)$-block-type set in Lemma 3 by a $2s(\Delta_D + 1)$-type set:

▶ **Lemma 4.** *Let (D, s, τ) be a* DDConC *instance and let $C \subseteq V(D)$ be a $2s(\Delta_D + 1)$-type set. If (D, s, τ) is a yes-instance, then there exists a solution $A^* \subseteq C^2$ for (D, s, τ), that is, $|A^*| \leq s$ and $\deg_{D+A^*}(v) \in \tau(v)$ for all $v \in V(D)$.*

Similarly, if there are no restrictions on the individual vertex degrees, that is, τ is the degree list function $\tau(v) = \{0, \ldots, n-1\}^2$ for all $v \in V(D)$, then we can replace the $2s(\Delta_D + 1)$-block-type set by a $2s(\Delta_D + 1)$-block set.

▶ **Lemma 5.** *Let (D, s, τ, Π) be a* DDConSeqC *instance where $\tau(v) = \{0, \ldots, n-1\}^2$ for all $v \in V(D)$ and let $C \subseteq V(D)$ be a $2s(\Delta_D + 1)$-block set. If (D, s, τ, Π) is a yes-instance, then there exists a solution $A^* \subseteq C^2$ for (D, s, τ, Π), that is, $|A^*| \leq s$ and $\sigma(D + A^*)$ fulfills Π.*

Lemma 3 implies a fixed-parameter algorithm by providing a reduced search space for possible solutions, namely any $2s(\Delta_D + 1)$-block-type set C: Simply try out all possibilities to insert at most s arcs with endpoints in C and check whether in one of the cases the degrees and the degree sequence of the resulting graph satisfy the requirements τ and Π. As $|C| \leq 2s(\Delta_D + 1) \cdot (\Delta_D + 1)^2(\Delta^*)^2$ and $\Delta^* \leq \Delta_D + s$, there are at most $O(2^{(2s(\Delta_D+1)^3(\Delta_D+s)^2)^2})$ possible subsets of arcs to insert. Altogether, this leads to the following theorem.

▶ **Theorem 6.** *If deciding Π is fixed-parameter tractable with respect to the maximum integer in the sequence, then* DDConSeqC *is fixed-parameter tractable with respect to (s, Δ_D).*

3.2 Bounding the solution size s polynomially in Δ^*

This subsection constitutes the major part of our framework. The rough overall scheme is analogous to the undirected case as described by Froese et al. [8]. By dropping the graph structure and solving a simpler problem-specific number problem on the degree sequence of the input digraph, we show how to solve DDConSeqC instances with "large" solutions provided that we can solve the associated number problem efficiently. The number problem is defined so as to simulate the insertion of arcs to a digraph on an integer tuple sequence. Note that inserting an arc increases the indegree of a vertex by one and increases the outdegree of another vertex by one. Inserting s arcs can thus be represented by increasing the tuple entries in the degree sequence by an overall value of s in each component. Formally, the corresponding number problem (abbreviated as #DDConSeqC) is defined as follows.

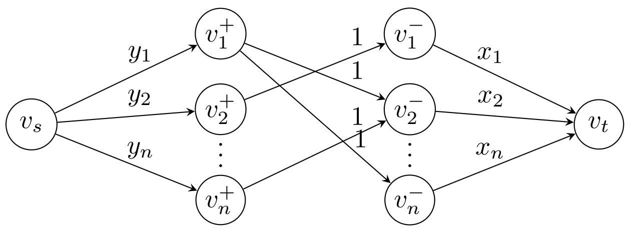

Figure 1 A flow network as described in Construction 8. For each vertex v_i in the digraph D there are two vertices v_i^+ and v_i^-. We connect a vertex v_i^+ to a vertex v_j^- if the arc (v_i, v_j) is not in D. Inserting the arc (v_i, v_j) is then represented by setting the flow on the arc (v_i^+, v_j^-) to one.

Numbers Only Digraph Degree Constraint Sequence Completion

Input: A sequence $\sigma = (c_1, d_1), \ldots, (c_n, d_n)$ of n nonnegative integer tuples, a positive integer s, a "tuple list function" $\tau\colon \{1, \ldots, n\} \to 2^{\{0,\ldots,r\}^2}$, and a sequence property Π.

Question: Is there a sequence $\sigma' = (c'_1, d'_1), \ldots, (c'_n, d'_n)$ such that $\sum_{i=1}^n c'_i - c_i = \sum_{i=1}^n d'_i - d_i = s$, $c_i \le c'_i$, $d_i \le d'_i$, and $(c'_i, d'_i) \in \tau(i)$ for all $1 \le i \le n$, and σ' fulfills Π?

If we plug the degree sequence of a digraph into #DDConSeqC, then an integer tuple (c'_i, d'_i) of a solution tells us to add $x_i := c'_i - c_i$ incoming arcs and $y_i := d'_i - d_i$ outgoing arcs to the vertex v_i. We call the tuples (x_i, y_i) *demands.* Having computed the demands, we can then try to solve our original DDConSeqC instance by searching for a set of arcs to insert that exactly fulfills the demands. Such an arc set, however, might not always exist. Hence, the remaining problem is to decide whether it is possible to realize the demands in the given digraph. The following lemma shows (using flow computations) that this is in fact always possible if the number s of arcs to insert is large compared to Δ^*.

▶ **Lemma 7.** *Let $D = (V = \{v_1, \ldots, v_n\}, A)$ be a digraph and let $x_1, \ldots, x_n$, $y_1, \ldots, y_n$, and Δ^* be nonnegative integers such that*

(I) $\Delta^* \le n - 1$,
(II) $\deg^-_D(v_i) + x_i \le \Delta^*$ *for all* $i \in \{1, \ldots, n\}$,
(III) $\deg^+_D(v_i) + y_i \le \Delta^*$ *for all* $i \in \{1, \ldots, n\}$,
(IV) $\sum_{i=1}^n x_i = \sum_{i=1}^n y_i =: s$, *and*
(V) $s > 2(\Delta^*)^2$.

Then, there exists an arc set $A' \subseteq V^2 \setminus A$ of size s such that for the digraph $D' := D + A'$ it holds $\deg_{D'}(v_i) = \deg_D(v_i) + (x_i, y_i)$ for all $v_i \in V$. Moreover, the set A' can be computed in $O(n^3)$ time.

Proof. The proof is based on a flow network which we construct such that the corresponding maximum flow yields the set A' of arcs to be inserted in D in order to obtain our target digraph D'.

▶ **Construction 8.** *We build a flow network $N = (V_N, A_N)$ according to the following steps.*

- *Add a* source *vertex v_s and a* sink *vertex v_t to N;*
- *for each vertex $v_i \in V$, add two vertices v_i^+, v_i^- to N;*
- *for each $i \in \{1, \ldots, n\}$, insert the arc (v_s, v_i^+) with capacity y_i;*
- *for each $i \in \{1, \ldots, n\}$, insert the arc (v_i^-, v_t) with capacity x_i;*
- *for each $(v_i, v_j) \in V^2 \setminus A$ with $i \ne j$, insert the arc (v_i^+, v_j^-) with capacity one.*

The network N contains $|V_N| \in O(n)$ vertices and $|A_N| \in O(n^2 - m)$ arcs (since $m \le n^2 - n$, we also have $|A_N| \in \Omega(n)$) and can be constructed in $O(n^2)$ time. See Figure 1 for an illustration. Inserting an arc (v_i, v_j) in D corresponds to sending flow from v_i^+ to v_j^-. Since, by definition, each vertex v_i^+ will only receive at most y_i flow from v_s and each vertex v_i^- will send at most x_i flow to v_t, we cannot insert more than s arcs (Condition (IV)).

We claim that for $s > 2(\Delta^*)^2$ (Condition (V)), the maximum flow in the network is indeed s. To see this, let $V_N^+ := \{v_i^+ \in V_N \mid i \in \{1,\dots,n\}\}$ and let $V_N^- := \{v_i^- \in V_N \mid i \in \{1,\dots,n\}\}$. In the following, a vertex $v_i^+ \in V_N^+$ ($v_j^- \in V_N^-$) is called *saturated* with respect to a flow $f\colon A_N \to \mathbb{R}^+$, if $f(v_s, v_i^+) = y_i$ ($f(v_j^-, v_t) = x_j$). Suppose that the maximum flow f has a value less than s. Then, there exist non-saturated vertices $v_i^+ \in V_N^+$ and $v_j^- \in V_N^-$. Let $X \subseteq V_N^-$ be the vertices to which v_i^+ has an outgoing arc in the residual graph and let $Y \subseteq V_N^+$ be the vertices which have an outgoing arc to v_j^- in the residual graph. Observe that $\deg_N^+(v_i^+) = n - 1 - \deg_D^+(v_i)$ and $\deg_N^-(v_j^-) = n - 1 - \deg_D^-(v_j)$. Consequently, $|X| > n - 1 - \deg_D^+(v_i) - y_i \geq n - 1 - \Delta^*$ holds due to Condition (III). Since v_i^+ is not saturated, we know that $|X| \geq n - \Delta^* \geq 1$ (due to Condition (I)). By the same reasoning (using Conditions (II) and (I)) it follows that $|Y| \geq n - \Delta^* \geq 1$.

Remember that f is a flow of maximum value. Hence, each vertex in X and each vertex in Y is saturated. Otherwise, there would be an augmenting path in the residual graph, contradicting our assumption of f being maximal. If a vertex $x \in X$ would receive flow from a vertex $y \in Y$, then this implies a backward arc in the residual graph resulting in an augmenting path $v_s \to v_i^+ \to x \to y \to v_j^- \to v_t$, again contradicting our maximality assumption for f. Thus, we can conclude that all the flow that goes into X has to come from the remaining vertices in $V_N^+ \setminus (Y \cup \{v_i^+\})$. This set has size at most $n - |Y| \leq n - (n - \Delta^*) = \Delta^*$. But since $y_\ell \leq \Delta^*$ for all $\ell \in \{1,\dots,n\}$ (by Condition (III)), those Δ^* vertices can cover at most a flow of value $(\Delta^*)^2$ and hence,

$$\sum_{v_i^- \in X} x_i \leq \sum_{v_i^+ \in V_N^+ \setminus (Y \cup \{v_i^+\})} y_i \leq (\Delta^*)^2. \tag{1}$$

Since X is saturated, and since also $x_\ell \leq \Delta^*$ holds for all $\ell \in \{1,\dots,n\}$ (Condition (II)), we obtain from Condition (IV)

$$\begin{aligned} s = \sum_{i=1}^{n} x_i = \sum_{v_i^- \in X} x_i + \sum_{v_i^- \in V_N^- \setminus X} x_i &\overset{(1)}{\leq} (\Delta^*)^2 + \sum_{v_i^- \in V_N^- \setminus X} \Delta^* \\ &= (\Delta^*)^2 + |V_N^- \setminus X| \cdot \Delta^* = (\Delta^*)^2 + (n - |X|) \cdot \Delta^* \\ &\leq (\Delta^*)^2 + \Delta^* \cdot \Delta^*. \end{aligned}$$

This contradicts $s > 2(\Delta^*)^2$ (Condition (V)) and hence proves the claim.

Now, let f be a maximum flow in N (computable in $O(|V_N||E_N|) = O(n(n^2 - m))$ time [21]) and let $A' := \{(v_i, v_j) \in V^2 \mid f((v_i^+, v_j^-)) = 1\}$ and note that $|A'| = s$ and $A' \cap A = \emptyset$. Clearly, for the digraph $D' := D + A'$ it holds $\deg_{D'}(v_i) = \deg_D(v_i) + (x_i, y_i)$ for all $v_i \in V$. ◀

We remark that similar flow-constructions as given in the proof above have been used before [9, 6]. The difference here is that we actually argue about the size of the flow and not only about polynomial-time solvability. Consequently, our proof uses different arguments.

With Lemma 7 we have the key which allows us to transfer solutions of #DDConSeqC to solutions of DDConSeqC. The following lemma is immediate.

▶ **Lemma 9.** *Let $I := (D = (V, A), s, \tau, \Pi)$ with $V = \{v_1, \dots, v_n\}$ be an instance of* DDConSeqC *with $s > 2(\Delta^*)^2$. If there exists an s' with $2(\Delta^*)^2 < s' \leq s$ such that $I' := (\deg_D(v_1), \dots, \deg_D(v_n), s', \tau', \Pi)$ with $\tau'(i) := \tau(v_i)$ for all $v_i \in V$ is a yes-instance of* #DDConSeqC*, then also I is a yes-instance of* DDConSeqC.

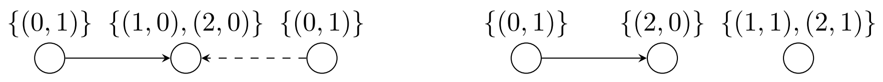

Figure 2 Two example instances of DDConC with $s = 1$. The left instance is solvable by inserting the (dashed) arc from the right vertex to the middle vertex. The right instance is a no-instance since one cannot add an outgoing arc to the left vertex or to the middle vertex but one has to add an incoming arc to the right vertex (loops are not allowed).

We now have all ingredients for our first main result, namely transferring fixed-parameter tractability with respect to the combined parameter (s, Δ^*) to fixed-parameter tractability with respect to the single parameter Δ^*, provided that #DDConSeqC is fixed-parameter tractable with respect to the largest possible integer ξ in the output sequence. The idea is to search for large solutions based on Lemma 9 using #DDConSeqC. If there are no large solutions (that is, $s \leq 2(\Delta^*)^2$), then we run an FPT-algorithm with respect to (s, Δ^*).

▶ **Theorem 10.** *If* DDConSeqC *is fixed-parameter tractable with respect to* (s, Δ^*) *and* #DDConSeqC *is fixed-parameter tractable with respect to the largest possible integer* ξ *in the output sequence, then* DDConSeqC *is fixed-parameter tractable with respect to* Δ^*.

Our second main result allows to transfer a polynomial-size problem kernel with respect to (s, Δ^*) to a polynomial-size problem kernel with respect to Δ^* if #DDConSeqC is polynomial-time solvable. The proof is analogous to the proof of Theorem 10.

▶ **Theorem 11.** *If* DDConSeqC *admits a problem kernel containing* $g(s, \Delta^*)$ *vertices computable in* $p(n)$ *time and* #DDConSeqC *is solvable in* $q(n)$ *time for polynomials* p *and* q, *then* DDConSeqC *admits a problem kernel with* $g(2(\Delta^*)^2, \Delta^*)$ *vertices computable in* $O(s \cdot q(n) + p(n))$ *time.*

4 Applications

In the following, we show how the framework described in Section 3 can be applied to three special cases of DDConSeqC. These special cases naturally extend known problems on undirected graphs to the digraph setting.

4.1 Digraph Degree Constraint Completion

In this section, we investigate the NP-hard special case of DDConSeqC[2] where the property Π allows any possible degree sequence, see Figure 2 for two illustrating examples.

Digraph Degree Constraint Completion (DDConC)

Input: A digraph $D = (V, A)$, a positive integer s, and a "degree list function" $\tau\colon V \to 2^{\{0,\ldots,r\}^2}$.

Question: Is it possible to obtain a digraph D' by inserting at most s arcs in D such that $\deg_{D'}(v) \in \tau(v)$ for all $v \in V$?

DDConC is the directed (completion) version of the well-studied undirected Degree Constraint Editing problem [18, 10] for which problem kernel with $O(r^5)$-vertices is

[2] This special case was investigated more specifically in the Bachelor thesis of Koseler [15] (online available).

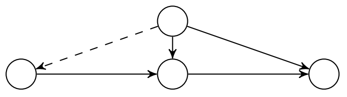

Figure 3 Example instance of DDSeqC. Inserting the dashed arc in the input digraph (solid arcs) with degree sequence $\{(0,1),(0,2),(2,0),(2,1)\}$ yields a digraph with the given target sequence σ.

known [8]. We subsequently transfer the polynomial-size problem kernel for the undirected case to a polynomial-size problem kernel for DDConC with respect to Δ^*. Note that the parameter Δ^* is clearly at most r. Since it is trivial to decide Π in this case, we obtain fixed-parameter tractability of DDConC with respect to (s, Δ_D) due to Theorem 6, which is based on a bounded search space, namely a $2s(\Delta_D+1)$-type set (see Definition 1 and Lemma 4). We further strengthen this result by removing all vertices that are not in the $2s(\Delta_D+1)$-type set and adjusting the degree list function τ appropriately. Lemma 4 then yields the correctness of this approach resulting in a polynomial-size problem kernel with respect to (s, Δ^*).

▶ **Theorem 12.** DDConC *admits a problem kernel containing $O(s(\Delta^*)^3) \subseteq O(sr^3)$ vertices. It is computable in $O(m + |\tau| + r^2)$ time.*

The goal now is to use our framework (Theorem 11) to transfer the polynomial-size kernel with respect to (s, Δ^*) to a polynomial-size kernel with respect to Δ^*. To this end, we show that the corresponding number problem (#DDConC) is polynomial-time solvable. Here, #DDConC is the special case of #DDConSeqC without the sequence property Π. #DDConC can be solved in pseudo-polynomial time by a dynamic programming algorithm. Note that pseudo-polynomial time is sufficient for our purposes since all occurring numbers will be bounded by $O(n^2)$ when creating the #DDConC instance from the given DDConC instance. (In fact, we conjecture that #DDConC is weakly NP-hard and a reduction from Partition should be possible as in the case for #DDA in Section 4.3, Theorem 19.)

▶ **Lemma 13.** #DDConC *is solvable in $O(n(sr)^2)$ time.*

Combining Theorem 12 and Lemma 13 yields the following corollary of Theorem 11.

▶ **Corollary 14.** DDConC *admits a problem kernel containing $O((\Delta^*)^5) \subseteq O(r^5)$ vertices. It is computable in $O(m + ns^3r^2)$ time.*

4.2 Digraph Degree Sequence Completion

In this section, we investigate the NP-hard special case of DDConSeqC[3] where τ does not restrict the allowed degree of any vertex and Π is fulfilled by exactly one specific degree sequence σ (see Figure 3 for an example). The undirected problem variant is studied by Golovach and Mertzios [11].

Digraph Degree Sequence Completion (DDSeqC)

Input: A digraph $D = (V, A)$, a digraph degree sequence σ containing $|V|$ integer tuples.

Question: Is it possible to obtain a digraph D' by inserting arcs in D such that $\sigma(D') = \sigma$?

[3] Although not stated explicitly, the NP-hardness follows from the proof of Theorem 3.2 of the Bachelor thesis of Millani [19] (online available) as the construction therein allows for only one feasible target degree sequence.

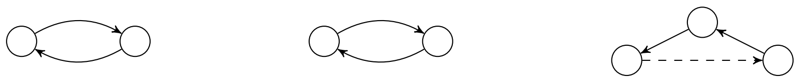

Figure 4 Example instance of DDA. The input digraph with three components (solid arcs) is 1-anonymous since there is only one vertex with degree $(0,1)$. By inserting the dashed arc, the digraph becomes 7-anonymous since all vertices have degree $(1,1)$.

For DDSeqC, the parameter Δ^* is by definition equal to Δ_σ. Moreover, note that the number s of arcs to insert (if possible) is determined by the target sequence σ by $s := \sum_{(c,d)\in\sigma} c - \sum_{v\in V(D)} \deg^-_D(v)$. We henceforth assume that

$$s = \sum_{(c,d)\in\sigma} c - \sum_{v\in V(D)} \deg^-_D(v) = \sum_{(c,d)\in\sigma} d - \sum_{v\in V(D)} \deg^+_D(v) \geq 0$$

holds since otherwise we have a trivial no-instance.

Since deciding Π (that is, deciding whether $\sigma(D') = \sigma$) can be done in polynomial time, we immediately obtain fixed-parameter tractability of DDSeqC with respect to (s, Δ_D) due to Theorem 6. We further strengthen this result by developing a polynomial-size problem kernel for DDSeqC with respect to (s, Δ_σ). The kernelization is inspired by the $O(s\Delta_\sigma^2)$-vertex problem kernel for the undirected problem by Golovach and Mertzios [11]. The main idea is to only keep the vertices of a $2s(\Delta_D + 1)$-block set (see Definition 1) together with some additional "dummy" vertices and to adjust the digraph degree sequence σ properly.

▶ **Theorem 15.** DDSeqC *admits a problem kernel containing $O(s\Delta_\sigma^3)$ vertices computable in $O(n + m + \Delta_\sigma^2)$ time.*

The corresponding number problem #DDSeqC is the special case of #DDConSeqC asking for the specific target sequence σ. #DDSeqC can be solved in polynomial time by finding perfect matchings in an auxiliary graph.

▶ **Lemma 16.** #DDSeqC *is solvable in $O(n^{2.5})$ time.*

Combining Theorem 15 and Lemma 16 yields the following corollary of Theorem 11.

▶ **Corollary 17.** DDSeqC *admits a problem kernel containing $O(\Delta_\sigma^5)$ vertices. It is computable in $O(sn^{2.5})$-time.*

4.3 Degree Anonymity

We extend the definition of Degree Anonymity in undirected graphs due to Liu and Terzi [17] to digraphs and obtain the following NP-hard problem [19] (Figure 4 presents an example):

Digraph Degree Anonymity (DDA)
Input: A digraph $D = (V, A)$ and two positive integers k and s.
Question: Is it possible to obtain a digraph D' by inserting at most s arcs in D such that D' is k-anonymous, that is, for every vertex $v \in V$ there are at least $k - 1$ other vertices in D' with degree $\deg_{D'}(v)$?

The (parameterized) complexity as well as the (in-)approximability of the undirected version called Degree Anonymity are well-studied [5, 14, 3]. There also exist many heuristic approaches to solve the undirected version [4, 13]. Notably, our generic approach shown in Section 3.2 originates from a heuristic of Liu and Terzi [17] for Degree Anonymity. Later, Hartung et al. [14] used this heuristic to prove that "large" solutions of Degree Anonymity can be found in polynomial time and Froese et al. [8] extended this approach to a more general class of problems. The property Π (that is, k-anonymity) can clearly be checked for a given input digraph degree sequence in polynomial time. Hence, Theorem 6 yields fixed-parameter tractability of DDA with respect to (s, Δ_D). Again, we develop a polynomial-size problem kernel with respect to (s, Δ_D). Somewhat surprisingly, we cannot transfer this problem kernel to a problem kernel with respect to Δ^* since we are not able to solve the corresponding number problem in polynomial time. In fact, we will show that it is at least weakly NP-hard.

We first provide a problem kernel based on Lemma 5 in a similar fashion as in the proof of Theorem 15: We keep a $2s(\Delta_D + 1)$-block set C in the kernel and remove all other vertices. In order to not change the degrees of the vertices we kept, we introduce "dummy" vertices that will have a very high degree so that there is no interference with the vertices we kept. The approach is inspired by the polynomial-size problem kernel of Hartung et al. [14].

▶ **Theorem 18.** DDA *admits a problem kernel containing $O(\Delta_D^5 s)$ vertices. It is computable in $O(\Delta_D^{10} s^2 + \Delta_D^3 sn)$ time.*

In contrast to both number problems in Sections 4.1 and 4.2, we were unable to find a polynomial-time algorithm for the number problem for DDA, which is the special case of #DDConSeqC asking for a k-anonymous target sequence. We can show that #DDA is weakly NP-hard by a polynomial-time many-one reduction from Partition.

▶ **Theorem 19.** #DDA *is (weakly) NP-hard even if $k = 2$.*

Note that the hardness from Theorem 19 does not translate to instances of #DDA originating from digraph degree sequences because in such instances all numbers in the input sequence σ and also in the output sequence σ' are bounded by $n - 1$ where n is the number of tuples in σ. Since there are pseudo-polynomial-time algorithms for Partition, Theorem 19 leaves open whether #DDA is strongly NP-hard or can be solved in polynomial time for instances originating from digraphs.

To again apply our framework (Theorem 10), we show that #DDA is at least fixed-parameter tractable with respect to the largest possible integer ξ in the output sequence. To this end, we develop an integer linear program that contains at most $O(\xi^4)$ integer variables and apply the a famous result due to Lenstra [16].

▶ **Theorem 20.** #DDA *is fixed-parameter tractable with respect to the largest possible integer ξ in the output sequence.*

Combining Theorems 6, 10, and 20 yields fixed-parameter tractability for DDA with respect to Δ^*. Hartung et al. [14] showed fixed-parameter tractability with respect to Δ_G in the undirected setting. This result was based on showing that $\Delta^* \leq \Delta_G^2 + 5\Delta_G^2 + 2$. In the directed setting, however, we can only show that $\Delta^* \leq 4k(\Delta_D + 2)^2$.

▶ **Lemma 21.** *Let D be a digraph and let S be a minimum size arc set such that $D + S$ is k-anonymous. Then the maximum degree in $D + S$ is at most $4k(\Delta_D + 2)^2 + \Delta_D$.*

Consequently, combining Theorems 6, 10, 20, and Lemma 21, we obtain the following.

▶ **Corollary 22.** DDA *is fixed-parameter tractable with respect to Δ^* and (k, Δ_D).*

It remains open whether DDA is fixed-parameter tractable with respect to Δ_D. We remark that the problems DDConC and DDSeqC are both NP-hard for $\Delta_D = 3$. This follows from an adaption of the construction given by Millani [19, Theorem 3.2].

5 Conclusion

We proposed a general framework for digraph degree sequence completion problems and demonstrated its wider applicability in case studies. Somewhat surprisingly, the presumably more technical case of digraphs allowed for some elegant tricks (based on flow computations) that seem not to work for the presumably simpler undirected case. Once having established the framework (see Section 3), the challenges then associated with deriving fixed-parameter tractability and kernelizability results usually boil down to the question for fixed-parameter tractability and (pseudo-)polynomial-time solvability of a simpler problem-specific number problem. While in most cases we could develop polynomial-time algorithms solving these number problems, in the case of Digraph Degree Anonymity the polynomial-time solvability of the associated number problem remains open. Moreover, a widely open field is to attack weighted versions of our problems. Finally, we believe that due to the fact that many real-world networks are inherently directed (e.g., representing relations such as "follower", "likes", or "cites") further studies (e.g., exploiting special digraph properties) of digraph degree sequence completion problems are desirable.

References

1 Jørgen Bang-Jensen, András Frank, and Bill Jackson. Preserving and increasing local edge-connectivity in mixed graphs. *SIAM Journal on Discrete Mathematics*, 8(2):155–178, 1995.

2 Jørgen Bang-Jensen, Jing Huang, and Xuding Zhu. Completing orientations of partially oriented graphs. *CoRR abs/1509.01301*, 2015.

3 Cristina Bazgan, Robert Bredereck, Sepp Hartung, André Nichterlein, and Gerhard J. Woeginger. Finding large degree-anonymous subgraphs is hard. *Theoretical Computer Science*, 622:90–110, 2016.

4 Jordi Casas-Roma, Jordi Herrera-Joancomartí, and Vicenç Torra. An algorithm for k-degree anonymity on large networks. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM'13)*, pages 671–675. ACM, 2013.

5 Sean Chester, Bruce Kapron, Gautam Srivastava, and S. Venkatesh. Complexity of social network anonymization. *Social Network Analysis and Mining*, 3(2):151–166, 2013.

6 Marek Cygan, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Ildikó Schlotter. Parameterized complexity of eulerian deletion problems. *Algorithmica*, 68(1):41–61, 2014.

7 Frederic Dorn, Hannes Moser, Rolf Niedermeier, and Mathias Weller. Efficient algorithms for eulerian extension and rural postman. *SIAM Journal on Discrete Mathematics*, 27(1):75–94, 2013.

8 Vincent Froese, André Nichterlein, and Rolf Niedermeier. Win-win kernelization for degree sequence completion problems. *Journal of Computer and System Sciences*, 82(6):1100–1111, 2016.

9 D. Gale. A theorem on flows in networks. *Pacific Journal of Mathematics*, 7:1073–1082, 1957.

10 Petr A. Golovach. Editing to a graph of given degrees. *Theoretical Computer Science*, 591:72–84, 2015.

11 Petr A. Golovach and George B. Mertzios. Graph editing to a given degree sequence. In *Proceedings of the 11th International Computer Science Symposium in Russia (CSR'16)*, volume 9691 of *LNCS*, pages 177–191. Springer, 2016.

12 Gregory Gutin and Anders Yeo. Some parameterized problems on digraphs. *Computer Journal*, 51(3):363–371, 2008.

13 Sepp Hartung, Clemens Hoffmann, and André Nichterlein. Improved upper and lower bound heuristics for degree anonymization in social networks. In *Proceedings of the 13th International Symposium on Experimental Algorithms (SEA'14)*, volume 8504 of *LNCS*, pages 376–387. Springer, 2014.

14 Sepp Hartung, André Nichterlein, Rolf Niedermeier, and Ondřej Suchý. A refined complexity analysis of degree anonymization in graphs. *Information and Computation*, 243:249–262, 2015.

15 Marcel Koseler. Kernelization for degree-constraint editing on directed graphs. Bachelor thesis, TU Berlin, November 2015. URL: `http://fpt.akt.tu-berlin.de/publications/theses/BA-marcel-koseler.pdf`.

16 Hendrik W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983.

17 Kun Liu and Evimaria Terzi. Towards identity anonymization on graphs. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD'08, pages 93–106. ACM, 2008.

18 Luke Mathieson and Stefan Szeider. Editing graphs to satisfy degree constraints: A parameterized approach. *Journal of Computer and System Sciences*, 78(1):179–191, 2012.

19 Marcelo Garlet Millani. Algorithms and complexity for degree anonymization in directed graphs. Bachelor thesis, TU Berlin, March 2015. URL: `http://fpt.akt.tu-berlin.de/publications/theses/BA-marcelo-millani.pdf`.

20 Hannes Moser and Dimitrios M. Thilikos. Parameterized complexity of finding regular induced subgraphs. *Journal of Discrete Algorithms*, 7(2):181–190, 2009.

21 James B. Orlin. Max flows in $o(nm)$ time, or better. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC'13)*, pages 765–774. ACM, 2013.

22 Mathias Weller, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. On making directed graphs transitive. *Journal of Computer and System Sciences*, 78(2):559–574, 2012.