Effective and Efficient Data Reduction for the Subset Interconnection Design Problem*

Jiehua Chen[1], Christian Komusiewicz[1], Rolf Niedermeier[1], Manuel Sorge[1], Ondřej Suchý[2], and Mathias Weller[3]

[1] Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, {jiehua.chen, christian.komusiewicz, rolf.niedermeier, manuel.sorge}@tu-berlin.de
[2] Department of Theoretical Computer Science, Czech Technical University in Prague, ondrej.suchy@fit.cvut.cz
[3] Département Informatique, LIRMM, mathias.weller@lirmm.fr

Abstract. The NP-hard Subset Interconnection Design problem is motivated by applications in designing vacuum systems and scalable overlay networks. It has as input a set V and a collection of subsets $V_1, V_2, \ldots, V_m$, and asks for a minimum-cardinality edge set E such that for the graph $G = (V, E)$ all induced subgraphs $G[V_1], G[V_2], \ldots, G[V_m]$ are connected. It has also been studied under the name Minimum Topic-Connected Overlay. We study Subset Interconnection Design in the context of polynomial-time data reduction rules that preserve optimality. Our contribution is threefold: First, we point out flaws in earlier polynomial-time data reduction rules. Second, we provide a fixed-parameter tractability result for small subset sizes and tree-like output graphs. Third, we show linear-time solvability in case of a constant number m of subsets, implying fixed-parameter tractability for the parameter m. To achieve our results, we elaborate on polynomial-time data reduction rules (partly "repairing" previous flawed ones) which also may be of practical use in solving Subset Interconnection Design.

1 Introduction

We study relevant tractable cases of the following NP-complete decision problem:

Subset Interconnection Design (SID)
Input: A hypergraph $H = (V, \mathcal{F})$, $k \in \mathbb{N}$.
Question: Is there a graph $G = (V, E)$ such that $|E| \leq k$ and for each $F \in \mathcal{F}$ the induced subgraph $G[F]$ is connected?

* JC was supported by Studienstiftung des Deutschen Volkes, MS and MW were supported by Deutsche Forschungsgemeinschaft (projects NI 369/12 and NI 369/9), and part of the work of OS and MW was done while they were affiliated with TU Berlin. This manuscript is to appear in *Proceedings of the 24th International Symposium on Algorithms and Computation (ISAAC 2013)*, Hong Kong, Hong Kong, December 2013.

Throughout this work, we refer to graphs G in which $G[F]$ is connected for each $F \in \mathcal{F}$ as *solutions*. Solutions with a minimum number of edges are called *optimal*. Although we present our results for the decision version of the problem, our positive algorithmic results can be easily adapted to its optimization version.

SID has applications in the design of vacuum systems [5, 6], in the design of scalable overlay networks [2, 11, 14], in the design of reconfigurable interconnection networks [7, 8], and in inferring a most likely social network [1]. Indeed, the respective research communities seemed largely unaware of each other's work, for instance leading to multiple NP-hardness proofs. Du [4] seemed to be the first to have formally defined the problem and claimed NP-hardness; to the best of our knowledge, the first published NP-hardness proof is due to Du and Miller [6]. SID has been independently studied under the name Minimum Topic-Connected Overlay by the "scalable overlay networks community" [2, 11, 14] and under the name Interconnection Graph Problem by the "reconfigurable interconnection systems community" [7, 8]. Moreover, the "social network inference community" [1], who additionally imposes edge costs, refers to this more general problem as Network Inference. The term "topic-connected" in Minimum Topic-Connected Overlay refers to the desired property of overlay networks that agents interested in some particular topic should be able to inform each other about updates concerning this topic without involving other agents [14].

Our main focus is on the problem-specific parameters "size $d := \max_{F \in \mathcal{F}} |F|$ of the largest hyperedge" and "number m of hyperedges" in the given hypergraph H. We perform a parameterized complexity analysis with respect to these parameters. Notably, we always have $d \leq k + 1$, where k is the number of edges of the constructed solution. In particular, our core working machinery is the development of numerous polynomial-time data reduction rules, thereby extending and improving some previous work. We use n to denote the number $|V|$ of vertices in the input hypergraph and $|H|$ to denote $\sum_{F \in \mathcal{F}} |F|$.

Previous results. As mentioned before, SID has been independently studied in different communities. Several NP-hardness proofs have appeared [2, 6, 7].[4] NP-hardness even holds for hypergraphs with $d = 3$ [8, 11], while $d \leq 2$ allows for polynomial-time solvability [11]. There also has been intense study of the polynomial-time approximability, providing various logarithmic-factor approximation algorithms [1, 2, 11] and inapproximability results (implying that logarithmic-factor approximation algorithms are optimal) [1, 11]. The currently best exact algorithm for SID has a running time of $O(n^{2k}/4^k + n^2)$ [11]. In addition, in a series of papers it has been shown that SID can be solved in polynomial time if $2 \leq m \leq 4$ [4, 15, 16]. A variant of SID where the edges incur costs and where the solution is restricted to be a tree has been studied in the context of communication network design; three variations of this tree-construction problem have been shown polynomial-time solvable [12]. Finally, we mention in passing that in the context of overlay networks it is of specific interest to search

[4] The reduction in [2] actually only shows NP-hardness of the problem aiming to minimize the maximum degree of a solution.

for solutions with small maximum and small average vertex degree [2, 14]; the latter is achieved by SID.

Our contributions. We start by revealing a serious bug in "plausible" data reduction rules (two very similar rules) used in previous work [8, 11], constructing a counterexample showing their incorrectness. Based on this, we provide refined and completely new data reduction rules, assuring their correctness and effectiveness. Almost all of our data reduction rules work in a parameter-independent fashion. Making decisive use of the developed data reduction rules, we show that SID can be solved in $d^{O(df)} \cdot \text{poly}(|H|)$ time, where f denotes the size of a minimum feedback edge set of an optimal solution G, that is, the minimum number of edges whose removal makes G acyclic. Our result shows that SID becomes tractable if the solution is required to be almost a tree (compare this with the tree requirement in related work [12]). Furthermore, a simple calculation shows that whenever $f \leq (n-1)/9d$ the exponential term in our algorithm is smaller than the one in the $O(n^{2k}/4^k + n^2)$-time algorithm given by Hosoda et al. [11]. In case that $d \leq 4$ we further show that SID can be reduced in polynomial time to an equivalent instance of $O(f)$ vertices, known as "polynomial-size problem kernel" in parameterized algorithmics. Finally, improving and generalizing previous work [4, 15, 16], we show that SID can be solved in linear time if the input hypergraph contains only a constant number of hyperedges. This implies that SID is fixed-parameter tractable with respect to the parameter m. Due to lack of space, most proofs are deferred to a full version of the paper.

2 Preliminaries

The concept of parameterized complexity was pioneered by Downey and Fellows [3] (see also [9, 13]). A parameterized problem is a language $L \subseteq \Sigma^* \times \Sigma^*$, where Σ is an alphabet. The second component is called the *parameter* of the problem. Typically, the parameter or the "combined" ones are non-negative integers. A parameterized problem L is *fixed-parameter tractable* (fpt) if there is an algorithm that decides whether $(x, k) \in L$ in $g(k) \cdot |x|^{O(1)}$ time, where g is an arbitrary computable function depending only on k. A core tool in the development of fixed-parameter algorithms is polynomial-time preprocessing by *data reduction* [10]. Here, the goal is to transform a given problem instance (x, k) in polynomial time into an equivalent instance (x', k') with parameter $k' \leq k$ such that the size of (x', k') is upper-bounded by some function g only depending on k. If this is the case, we call the instance (x', k') a (problem) *kernel* of size $g(k)$.

The data reduction is usually presented as a series of *reduction rules*, that is, polynomial-time algorithms that take as input an instance of some decision problem and also produces one as output. A reduction rule is *correct* if for each input instance I, the corresponding output instance of the rule is a yes-instance if and only if I is a yes-instance. Search tree algorithms can be described by *branching rules* that reduce one instance of a problem to several instances of the same problem; a branching rule is *correct* if the original instance is a yes-instance if and only if at least one of the constructed instances is a yes-instance.

Let V be a set and $\mathcal{F}$ be a family of subsets of V. We call $H = (V, \mathcal{F})$ a *hypergraph* with *vertex set* V and *hyperedge set* $\mathcal{F}$. Unless stated otherwise, we assume all hypergraphs to not contain singleton hyperedges, empty hyperedges or multiple copies of the same hyperedge since they are not meaningful for SID, and searching for and removing them can be done without increasing our running times. We call $v \in V$ and $F \in \mathcal{F}$ *incident* if $v \in F$. We denote by $\mathcal{F}(v)$ the set of all hyperedges that are incident with v. If $u, v \in V$ and $\mathcal{F}(v) \subseteq \mathcal{F}(u)$ then we say that u *covers* v. Vertices that cover each other are called *twins*; a maximal set of twins is called *twin class*. The *subhypergraph induced* by V' is the hypergraph $H[V'] := (V', \mathcal{F}')$ where $\mathcal{F}' = \{F \in \mathcal{F} \mid F \subseteq V'\}$. By *removing* a vertex v from H, we mean taking the hypergraph $H' = (V \setminus \{v\}, \{F \setminus \{v\} \mid F \in \mathcal{F}\})$. A *hyperwalk* is an alternating sequence of vertices and hyperedges starting and ending with a vertex and such that succeeding elements are incident with each other. A hypergraph is *connected* if there is a hyperwalk between every pair of vertices.

For graphs $G = (V, E)$ with vertex set V and edge set E, we use $E(G)$ to denote the edge set E of graph G. We denote by $G[V']$ the *subgraph of* G *induced* by V'. We also use $G - V'$ as a shorthand for $G[V \setminus V']$. The *feedback edge set* of a graph G is a minimum-size set of edges whose removal makes G a forest. If G is connected, then the size of a feedback edge set is $|E| - |V| + 1$.

3 Fundamental Observations

In this section, we show that a previously proposed data reduction rule for SID is incorrect. We also show some properties of SID and some data reduction rules that are used in our algorithms.

A very natural approach to identify edges of optimal solutions is to look for vertices u and v such that u covers v, that is, $\mathcal{F}(v) \subseteq \mathcal{F}(u)$. The following shows that degree-one vertices of the solution are adjacent to vertices that cover them.

Observation 1. If the hypergraph $H = (V, \mathcal{F})$ has a solution G such that some $u \in V$ has only one neighbor v in G, then v covers u.

It is thus tempting to devise a reduction rule that adds an edge between such vertices: creating a degree-one vertex should be optimal since every vertex needs at least one incident edge. Indeed, such a reduction rule was proposed for vertex pairs u, v that are twins, that is, they are in the same hyperedges [8], or where one covers the other [11]. The variant of these reduction rules that is applicable less often reads as follows.

Rule 1. If vertices u and v are twins, that is $\mathcal{F}(u) = \mathcal{F}(v)$, then remove u from H and decrease k by one.

Unfortunately, this rule is not correct, as a counterexample shows.

Lemma 1. *There is a yes-instance $(H = (V, \mathcal{F}), k)$ containing twins u and v such that Rule 1 applied to u and v yields a no-instance.*

Proof. Let $f \geq 3$ be an arbitrary integer. Consider the hypergraph $H = (V, \mathcal{F})$, with vertex set $V = \{u, v, a_1, \ldots, a_f, b_1, \ldots, b_f\}$ and hyperedge set $\mathcal{F}$ which is the union of the following sets of hyperedges:

$$\mathcal{F}_1 = \{\{a_i, b_i\} \mid i \in \{1, \ldots, f\}\},\ \mathcal{F}_2 = \{\{u, v, a_i, b_i\} \mid i \in \{1, \ldots, f\}\},$$
$$\mathcal{F}_3 = \{\{u, v, a_i, b_i, a_j\} \mid i, j \in \{1, \ldots, f\}, i \neq j\},\ \text{and}$$
$$\mathcal{F}_4 = \{\{u, v, a_i, b_i, b_j\} \mid i, j \in \{1, \ldots, f\}, i \neq j\}.$$

Note that the graph $G = (V, E)$ with $E := \mathcal{F}_1 \cup \{\{a_i, u\}, \{b_i, v\} \mid i \in \{1, \ldots, f\}\}$ is a solution for H containing $3f$ edges. Hence, $(H, 3f)$ is a yes-instance.

Now, let $(H' = (V', \mathcal{F}'), 3f - 1)$ be an instance that results from $(H, 3f)$ by applying Rule 1 to u and v, that is, removing u from H and decreasing the solution size by one. Then, $V' = V \setminus \{u\}$ and $\mathcal{F}'$ consists of the following hyperedges:

$$\mathcal{F}_1 = \{\{a_i, b_i\} \mid i \in \{1, \ldots, f\}\},\ \mathcal{F}'_2 = \{\{v, a_i, b_i\} \mid i \in \{1, \ldots, f\}\},$$
$$\mathcal{F}'_3 = \{\{v, a_i, b_i, a_j\} \mid i, j \in \{1, \ldots, f\}, i \neq j\},\ \text{and}$$
$$\mathcal{F}'_4 = \{\{v, a_i, b_i, b_j\} \mid i, j \in \{1, \ldots, f\}, i \neq j\}.$$

We show that every solution for H' has at least $3f$ edges and, thus, $(H', 3f - 1)$ is a no-instance. First, every solution for H' contains the f edges corresponding to the size-two hyperedges of $\mathcal{F}_1$. Furthermore, due to the hyperedges in $\mathcal{F}'_2$, for each $i \in \{1, \ldots, f\}$, either $\{v, a_i\}$ or $\{v, b_i\}$ is in any solution. By the symmetry between a_i and b_i in the created hypergraph, assume without loss of generality that an optimal solution contains the edge $\{v, b_i\}$ for all $i \in \{1, \ldots, f\}$. Now, let $G' = (V', E')$ be such a solution for H' and let $A_1 = \{a_i \mid \{v, a_i\} \notin E'\}$ be the set of a_is that are *not* adjacent to v in G' and let A_2 denote the remaining a_is. Now if $A_1 = \emptyset$, then G' contains at least $3f$ edges. We show that in case $A_1 \neq \emptyset$ the graph G' also has at least $3f$ edges. Assume that G' is optimal and that every optimal solution has at least $g > 0$ vertices in A_1. For every hyperedge $F = \{v, a_i, b_i, a_j\}$ with $a_j \in A_1$ and $i \in \{1, \ldots, f\} \setminus \{j\}$, G' has an edge between a_j and $\{v, a_i, b_i\}$ since $G'[F]$ is connected. Note that if G' contains the edge $\{b_i, a_j\}$, then we can replace this edge by $\{v, a_j\}$: The hyperedge F is the only hyperedge that contains $\{b_i, a_j\}$ and does not already induce a connected subgraph. Clearly, $G'[F]$ can also be made connected by adding $\{v, a_j\}$ instead. This implies an optimal solution with $g - 1$ vertices in A_1, contradicting our choice of g. Hence, G' contains no edges $\{b_i, a_j\}$ with $i \neq j$. Consequently, in order to make each $\{v, a_i, b_i, a_j\} \in \mathcal{F}'_4$ with $a_j \in A_1$ connected, there is an edge between a_i and a_j.

Hence, G' has $g \cdot (f - g)$ edges between A_1 and A_2, $\binom{g}{2}$ edges between vertices in A_1 and another $f - g$ edges between v and A_2. Altogether the total number of edges in G' is thus at least $2f + g \cdot (f - g) + \binom{g}{2} + f - g \geq 3f$. This implies that $(H', 3f - 1)$ is a no-instance. □

With some additional conditions, rules similar to Rule 1 are correct (Rules 2 to 4, 6, and 8 below). First, if a vertex u is adjacent to some vertex v covering u in an optimal solution, then there is an optimal solution that shifts some or all other edges incident with u to v.

Lemma 2. *Let u, v be two vertices in a hypergraph H with v covering u. If H has an optimal solution G containing the edge $\{u, v\}$, then H also has an optimal solution with u being adjacent only to v.*

The above lemma immediately implies the following reduction rule.

Rule 2. If hypergraph H contains vertices u, v such that v covers u and there is an optimal solution G containing the edge $\{u, v\}$, then remove u from H and decrease k by one.

Note that the correctness of Rule 2 together with Lemma 1 implies that there are instances in which twins or vertices that cover each other are *not* adjacent in any optimal solution.

In the counterexample to Rule 1, there are only two twins and they are contained in hyperedges of size five, that is, the size-five hyperedges containing these two vertices have three other "unrelated" vertices. In the following, we show that this is tight, that is, if in each hyperedge that contains some u, all except two unrelated vertices cover u, then the reduction rule is correct.

Rule 3. If there are vertices u and $v_1, \ldots, v_q$ such that $\mathcal{F}(u) \subseteq \mathcal{F}(v_i)$ for every $i \in \{1, \ldots, q\}$ and for each hyperedge $F \in \mathcal{F}(u)$ we have $|F| \leq q + 3$, then remove u from H and decrease k by one.

Lemma 3. *Rule 3 is correct and can be applied exhaustively in $O(n \cdot |H|)$ time.*

Proof (Sketch). Let $Q = \{v_1, \ldots, v_q\}$ and N the set of neighbors of u in an optimal solution G. If $N \cap Q \neq \emptyset$ then the correctness follows from Rule 2. Otherwise, if N contains a neighbor w of some $v \in Q$ in G, then removing $\{u, w\}$ and adding $\{u, v\}$ yields another optimal solution and we can apply Rule 2. If N contains no neighbor of any $v \in Q$ we obtain $|F \cap N| \leq 1$ for all $F \in \mathcal{F}(u)$ because of the size bound on F. Hence, removing all edges incident with u and adding to u a single edge to a vertex in Q does not disconnect any $F \in \mathcal{F}(u)$.

The running time proof is deferred to a full version of the paper. □

As a corollary of Lemma 3, we also obtain correctness of the following rule since it is a special case of Rule 3. This rule will be useful in the next section.

Rule 4. If there are two vertices u and v such that $\mathcal{F}(u) \subseteq \mathcal{F}(v)$ and $|F| \leq 4$ for each hyperedge $F \in \mathcal{F}(u)$, then remove u from H and decrease k by one.

Note that the condition $|F| \leq 4$ in Rule 4 is also tight in the sense that if u is incident with hyperedges of size at least five, this rule is not correct (Lemma 1).

4 Data Reduction Rules for Sparse Solutions

In this section, we present a set of reduction rules whose aim is to remove parts of the instance where optimal solutions can be identified in polynomial time. In particular, we aim at finding structures that either produce tree-like parts or long degree-two paths in the solution. We stress that our data reduction rules are applicable regardless of the structure of an optimal solution. We merely use the size of its feedback edge set to provide formal performance guarantees.

4.1 Problem Kernel for f and $d \leq 4$

We now describe how we can remove all but $O(f)$ vertices from a SID instance with $d \leq 4$ in $O(n \cdot m^3)$ time by using Rule 4 and an additional reduction rule. Basically, the parameter f upper-bounds the number of vertices that are in cycles and have degree at least three, while Rule 4 ensures that there are no degree-one vertices in solutions. To get an upper bound on the number of vertices, we also have to deal with long paths. This is the purpose of Rule 5, which is also needed in Section 4.2 to deal with larger hyperedges. Hence, this rule is more general than needed for $d \leq 4$.

Rule 5. Let $(H = (V, \mathcal{F}), k)$ be an instance of SID. If H contains a vertex set $P := \{p_0, \ldots, p_{2d}\}$ with incident hyperedge set $\mathcal{F}' := \bigcup_{p \in P} \mathcal{F}(p)$ such that

1. no $p_i \in P$ covers any $p_j \in P$ with $j \neq i$,
2. for each $F \in \mathcal{F}'$ we have $F \cap P = \{p_i, \ldots, p_j\}$ for some $0 \leq i \leq j \leq 2d$,
3. for each $F \in \mathcal{F}'$ with $F \cap \{p_0, p_{2d}\} = \emptyset$, and for every vertex $v \in F \setminus P$, there is a vertex $p \in P$ that covers v, and
4. there is no hyperedge $F \in \mathcal{F}$ such that $F \cap P = \{p_i\}$ for any $0 < i < 2d$,

then for every $F \in \mathcal{F}'$ with $F \cap \{p_0, p_{2d}\} = \emptyset$, remove all vertices in $F \setminus P$ from H and decrease k by their number. Furthermore, remove the vertices $p_2, \ldots, p_{2d-2}$ from H and decrease k by $2d - 2$.

Intuitively, Conditions 1 and 2 indicate that a solution for such a hypergraph contains a long path and Condition 3 ensures that all vertices not in the path can be attached to it in a simple way.

Next, we give two observations that we need in the correctness proof and in the analysis of the running time of Rule 5. The first observation is about the structure of the hyperedges along the presumed path containing P.

Observation 2. Let H be a hypergraph and $P \subseteq V$ as in Rule 5. For every $0 < i < 2d$ there is a hyperedge F_i^+ such that $p_{i-1} \notin F_i^+$ and $\{p_i, p_{i+1}\} \subseteq F_i^+$ and also a hyperedge F_i^- such that $\{p_{i-1}, p_i\} \subseteq F_i^-$ and $p_{i+1} \notin F_i^-$. Moreover, there is a hyperedge F_0^- such that $F_0^- \cap P = \{p_0\}$ and a hyperedge F_{2d}^+ such that $F_{2d}^+ \cap P = \{p_{2d}\}$.

The second observation provides a lower bound for the number of edges in solutions for connected subhypergraphs.

Observation 3. Let $H = (V, \mathcal{F})$ be a hypergraph and let G be a solution for H. If the subhypergraph $H[V']$ induced by a vertex subset $V' \subseteq V$ is connected, then $|E(G[V'])| \geq |V'| - 1$.

Using these observations we can prove the correctness of Rule 5.

Lemma 4. *Rule 5 is correct and it is possible to find an application of Rule 5 or to decide that it does not apply to the hypergraph in $O(m^3 d^3)$ time.*

We now derive an upper bound on the number of vertices in reduced instances. As mentioned before, we use Rule 5 in Section 4.2, where we also need a similar upper bound on the number of vertices. However, the preconditions of Rule 5

will be satisfied here by Rule 4 and later by a different rule. Hence, we introduce a "cleared"-notion for hypergraphs that will be ensured by these rules. To state our results conveniently, we first introduce another definition.

Definition 1. The *2-core* of a graph G is the uniquely defined induced subgraph of G with maximum number of vertices and minimum vertex-degree two.

Definition 2. We say that a hypergraph $H = (V, \mathcal{F})$ is *cleared* if there is an optimal solution G for H such that each vertex of degree at least two is in the 2-core of G and, furthermore, for each $P := \{p_0, \ldots, p_{2d}\}$ with $P \subseteq V$ and $\mathcal{F}' := \bigcup_{p \in P} \mathcal{F}(p)$ that satisfy Conditions 1, 2, and 3 of Rule 5, it holds that H and P also satisfy Condition 4.

It turns out that Rule 4 "clears hypergraphs":

Lemma 5. *Let $H = (V, \mathcal{F})$ be a hypergraph with $d \leq 4$ that is reduced with respect to Rule 4. Then, H is cleared.*

We now bound the size of reduced instances. We also use this bound in Section 4.2 and, hence, prove it in a slightly more general form than needed for $d \leq 4$.

Lemma 6. *Let (H, k) be a yes-instance of* SID *such that H is connected, cleared, and reduced with respect to Rule 5. Then, there is a solution $G = (V, E)$ for (H, k) such that the 2-core of G has at most $(9d - 1)(f - 1)$ vertices and, hence, at most $9d \cdot f$ edges.*

Using Lemmas 5 and 6, and combining them with the observation that hypergraphs that are reduced with Rule 4 have solutions without degree-one vertices, we now obtain that exhaustively applying Rules 4 and 5 yields a polynomial-size problem kernel for SID parameterized by the parameter f, when $d \leq 4$.

Theorem 1. *An instance of* SID *with $d \leq 4$ can be reduced to an equivalent one with at most $35(f - 1)$ vertices in $O(n \cdot m^3)$ time.*

4.2 A Fixed-Parameter Algorithm for f and d

Our polynomial-time data reduction in the last section does not generalize easily to arbitrary d, but, using an additional reduction rule, we can obtain the same vertex-bound of the 2-core of a solution. However, many degree-one vertices may still remain and it seems unclear how to remove them for $d \geq 5$.

Nevertheless, using the bounded 2-core in solutions, we obtain a branching algorithm with running time $O(d^{O(d \cdot f)} \cdot m^2 + n \cdot m^3 \cdot d^3)$. The algorithm first applies Rule 5 and Rule 6 (below) to simplify the structure of the solution that we are looking for. Then, we apply a branching rule that branches into $O(d^2)$ cases and finds at least one of the edges in the 2-core of a solution. If the branching rule does not apply, then an optimal solution can be found in polynomial time.

First, to obtain the bound on the 2-core, we replace Rule 4 with Rule 6 to clear the input hypergraph and to make Lemma 6 applicable.

Rule 6. Let $H = (V, \mathcal{F})$ be a hypergraph and $\{u, u_1, \ldots, u_\ell\} \in \mathcal{F}$ such that u covers each u_i. Then, remove the vertices $u_1, \ldots, u_\ell$ from H and decrease k by ℓ.

We use Rule 6 to replace Rule 4 in clearing hypergraphs.

Lemma 7. *Let $H = (V, \mathcal{F})$ be a hypergraph that is reduced with respect to Rule 6. Then, H is cleared.*

Now, Lemma 6 is applicable to hypergraphs that are reduced with respect to Rule 6 giving us that there is a solution with at most $9d \cdot f$ edges in the 2-core. Based on this lemma, we devise a branching algorithm for the parameter (d, f). This algorithm creates a search tree where at each node of the search tree the current instance consists of a hypergraph H, a partial solution G, and an integer k'. The task is to find a solution G' such that G' is a supergraph of G, all edges of G are within the 2-core of G', and the 2-core of G' has at most k' edges more than G. In order to obtain a search tree whose size depends only on d and f, we ensure that the search tree has depth at most $9d \cdot f$ and that the algorithm branches into at most $\binom{d}{2}$ cases in each step.

In the following, we assume that G and H are reduced with respect to Rule 6.

Branching Rule 1. Let F be a hyperedge of H such that $G[F]$ is disconnected and let $F_0 \subseteq F$ denote the vertices in F that have degree zero in G. Furthermore, $G[F]$ cannot be made connected by adding for each $u \in F_0$ an edge between u and some vertex $v \in F \setminus F_0$ that covers u. Then, branch into all possibilities to add an edge to $G[F]$, decreasing k by one.

Next, we show that, if Branching Rule 1 does not apply to any vertex, then we can solve the instance by greedily assigning the remaining vertices.

Lemma 8. *Let H be a hypergraph and let G be a graph such that there is a solution for H that is a supergraph of G and Branching Rule 1 does not apply to H and G. Then, an optimal solution for H can be computed in $O(n \cdot m)$ time.*

Combining all of the above, we arrive at the main result of this section.

Theorem 2. SID *can be solved in $O(d^{O(d \cdot f)} \cdot m^2 + n \cdot m^3 \cdot d^3)$ time.*

5 Data Reduction for Instances with Few Hyperedges

In this section, we show that SID is fixed-parameter tractable with respect to the number m of hyperedges. A previous fixed-parameter tractability result for this parameter relied on Rule 1 [11, Theorem 8] and is therefore incorrect. In order to restore this result, we need a slightly more involved rule whose correctness proof makes use of the following upper bound on the number of edges needed in the solution.

Lemma 9. *Every instance with $k \geq \binom{2^m}{2} + n$ is a yes-instance.*

The upper bound provided by Lemma 9 grows exponentially in the number of hyperedges. For many purposes, it would be practical to replace this exponential dependence by a polynomial function. However, we note that there are instances that require a solution with at least $n + 2^{\Omega(m)}$ edges (proof deferred).

Lemma 9 directly yields the following reduction rule.

Rule 7. If $k \geq \binom{2^m}{2} + n$, then answer "yes".

The following rule removes vertices from large twin classes.

Rule 8. Let H be an instance that is reduced with respect to Rule 7. If there is a twin class T in H with $|T| > 4^m + 7 \cdot 2^m + 1$, then remove an arbitrary vertex $v \in T$ from H and decrease k by one.

To prove the correctness of Rule 8, we need to show that there is a solution G that has the following property concerning its low-degree vertices.

Lemma 10. *Let $H = (V, \mathcal{F})$ be a hypergraph. There is a solution $G = (V, E)$ such that for each twin class T of H the graph G has*

1. *at most one vertex $t \in T$ that has degree-one neighbors, and*
2. *at most one degree-two vertex $t' \in T$.*

Now, the main idea for the proof of correctness of Rule 8 is to show that a solution G with $k < \binom{2^m}{2} + n$ edges for a hypergraph H cannot contain too many vertices of degree at least three. As a consequence, at least one vertex of T has degree one in G and can be removed safely.

Exhaustive application of Rule 7 and Rule 8 yields a problem kernel for SID parameterized by the number m of hyperedges. Moreover, this kernel can be computed in linear time.

Theorem 3. *An instance of* SID *can be reduced to an equivalent one of size at most $O(8^m \cdot m)$ in $O(|H|)$ time.*

6 Conclusion

Our work leads to a number of interesting tasks for future research: We left open the existence of a polynomial-size problem kernel for Subset Interconnection Design parameterized by the number m of hyperedges; we conjecture that there is none, however. Further, we did not resolve whether SID is fixed-parameter tractable with respect to the feedback edge set size of the solution *alone*. It would also be interesting to significantly improve on the straightforward exponential upper bound $2^{O(n^2)}$ when solving Subset Interconnection Design parameterized by the number n of vertices. It seems also promising to consider data reduction for the variant of Subset Interconnection Design that asks to minimize the maximum degree instead of the average degree (see Onus and Richa [14]). It is furthermore of practical interest to deal with edge weights for the constructed network [12]; our methods only cover the unweighted case. Given the numerous applications, an in-depth investigation of all relevant parameters motivated by real-world instances, that is, performing a parameter analysis for real-world instances, is promising from a practical and from a theoretical side.

Acknowledgment. We thank Peter Damaschke for stimulating discussions and for pointing us to the Subset Interconnection Designs problem.

References

[1] D. Angluin, J. Aspnes, and L. Reyzin. Inferring social networks from outbreaks. In *Proc. 21st ALT*, volume 6331 of *LNCS*, pages 104–118. Springer, 2010.

[2] G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg. Constructing scalable overlays for pub-sub with many topics. In *Proc. 26th PODC*, pages 109–118. ACM, 2007.

[3] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999.

[4] D.-Z. Du. An optimization problem on graphs. *Discrete Appl. Math.*, 14 (1):101 – 104, 1986.

[5] D.-Z. Du and D. F. Kelley. On complexity of subset interconnection designs. *J. Global Optim.*, 6(2):193–205, 1995.

[6] D.-Z. Du and Z. Miller. Matroids and subset interconnection design. *SIAM J. Discrete Math.*, 1(4):416–424, 1988.

[7] H. Fan and Y.-L. Wu. Interconnection graph problem. In *Proc. FCS 2008*, pages 51–55. CSREA Press, 2008.

[8] H. Fan, C. Hundt, Y.-L. Wu, and J. Ernst. Algorithms and implementation for interconnection graph problem. In *Proc. 2nd COCOA*, volume 5165 of *LNCS*, pages 201–210. Springer, 2008.

[9] J. Flum and M. Grohe. *Parameterized Complexity Theory.* Springer, 2006.

[10] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007.

[11] J. Hosoda, J. Hromkovič, T. Izumi, H. Ono, M. Steinová, and K. Wada. On the approximability and hardness of minimum topic connected overlay and its special instances. *Theor. Comput. Sci.*, 429:144–154, 2012.

[12] E. Korach and M. Stern. The clustering matroid and the optimal clustering tree. *Math. Program.*, 98(1-3):385–414, 2003.

[13] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.

[14] M. Onus and A. W. Richa. Minimum maximum-degree publish-subscribe overlay network design. *IEEE/ACM Trans. Netw.*, 19(5):1331–1343, 2011.

[15] T.-Z. Tang. An optimality condition for minimum feasible graphs. *Applied Mathematics - A Journal of Chinese Universities*, pages 24–21, 1989. In Chinese.

[16] Y. Xu and X. Fu. On the minimum feasible graph for four sets. *Applied Mathematics - A Journal of Chinese Universities*, 10:457–462, 1995.