POLYNOMIAL-TIME DATA REDUCTION FOR THE SUBSET INTERCONNECTION DESIGN PROBLEM[†]

JIEHUA CHEN[‡], CHRISTIAN KOMUSIEWICZ[‡], ROLF NIEDERMEIER[‡], MANUEL SORGE[‡], ONDŘEJ SUCHÝ[§], AND MATHIAS WELLER[¶]

Abstract. The NP-hard Subset Interconnection Design problem, also known as Minimum Topic-Connected Overlay, is motivated by numerous applications including the design of scalable overlay networks and vacuum systems. It has as input a finite set V and a collection of subsets $V_1, V_2, \ldots, V_m \subseteq V$, 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. We study Subset Interconnection Design in the context of polynomial-time data reduction rules that preserve the possibility to construct optimal solutions. Our contribution is threefold: First, we show the incorrectness of earlier polynomial-time data reduction rules. Second, we show linear-time solvability in case of a constant number m of subsets, implying fixed-parameter tractability for the parameter m. Third, we provide a fixed-parameter tractability result for small subset sizes and tree-like output graphs. To achieve our results, we elaborate on polynomial-time data reduction rules which also may be of practical use in solving Subset Interconnection Design.

Key words. NP-hard problem, fixed-parameter tractability, kernelization, combinatorial algorithms, preprocessing, hypergraph support

AMS subject classifications. 05C65, 68Q25, 68R10, 68W05

1. Introduction.

In many applications one wants to construct an "overlay graph" that interconnects each member of a given family of subsets of a ground set or, equivalently, that connects each hyperedge of a given hypergraph. We study an NP-complete problem in this context where one wants to construct an overlay graph containing few edges. The formal definition of the corresponding decision problem reads as follows.

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

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*. Two examples are shown in Figure 1. Although we present our results for the decision version of Subset Interconnection Design, our positive algorithmic results can be easily adapted to its optimization version.

[†]JC was supported by Studienstiftung des Deutschen Volkes. MS and MW were supported by Deutsche Forschungsgemeinschaft, projects DAPA (NI 369/12) and DARE (NI 369/11). OS was partly supported by Deutsche Forschungsgemeinschaft, project AREG (NI 369/9), and by the Czech Science Foundation, project 14-13017P. The main part of the work of OS and MW was done while they were affiliated with TU Berlin. An extended abstract of this article appeared under the title "Effective and Efficient Data Reduction for the Subset Interconnection Design Problem" in the *Proceedings of the 24th International Symposium on Algorithms and Computation (ISAAC 2013)*, volume 8283 of LNCS, Springer, pages 361–371. This manuscript is to appear in *SIAM Journal on Discrete Mathematics*.

[‡]Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany. ({jiehua.chen, christian.komusiewicz, rolf.niedermeier, manuel.sorge}@tu-berlin.de).

[§]Department of Theoretical Computer Science, Czech Technical University in Prague, Czech Republic. (ondrej.suchy@fit.cvut.cz).

[¶]Département Informatique, LIRMM, Montpellier University, France. (mathias.weller@lirmm.fr).

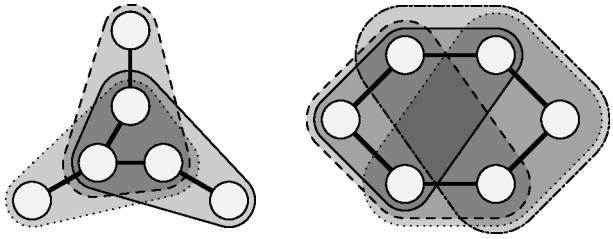

Figure 1: Two hypergraphs with optimal solutions. Vertices are drawn as white circles, hyperedges by grouping their incident vertices together inside a closed curve filled semi-transparently, and solution edges are drawn as thick lines.

Subset Interconnection Design is a fundamental problem concerning hypergraph and graph structures that has many applications. Indeed, Subset Interconnection Design has been studied in the context of designing vacuum systems [10, 11], scalable overlay networks [6, 18, 26], reconfigurable interconnection networks [13, 14], and, in variants, in the context of inferring a most likely social network [1], determining winners of combinatorial auctions [7] as well as drawing hypergraphs [3, 19, 21, 22]. The respective research communities seem largely unaware of each other's work, for instance leading to multiple NP-hardness proofs. To the best of our knowledge, the problem was first defined by Du [9] and the first NP-hardness proof was presented by Du and Miller [11]. SID has been independently studied under the names Minimum Topic-Connected Overlay by the "scalable overlay networks community" [6, 18, 26], Subset Interconnection Design by the "vacuum systems community" [10, 12], and Interconnection Graph Problem by the "reconfigurable interconnection systems community" [13, 14]. 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 [26]. The "social network inference community" [1], additionally imposing edge costs, refers to this more general problem as Network Inference. The "graph drawing community" [2, 3, 19, 20] refers to a solution as *support* and is mainly concerned with finding solutions with certain structure, for example, planar solutions. In further literature, solutions are also called *host graphs* [5] and *spanning graphs* [23].

Our main focus is on analyzing the influence of three problem-specific parameters on the computational complexity of SID. The parameters are "size $d := \max_{F \in \mathcal{F}} |F|$ of the largest hyperedge", "number $m := |\mathcal{F}|$ of hyperedges in the given hypergraph H", and "feedback edge number $\phi := k - n + 1$ of the solution" which is a measure of sparseness. Here, k is the number of edges of the constructed solution and n is the number $|V|$ of vertices in the input hypergraph H. We perform a parameterized complexity analysis with respect to these parameters showing that small parameters yield efficient algorithms. Our core working machinery is the development of numerous polynomial-time data reduction rules, thereby extending previous work.

Previous Work. As mentioned above, SID has been independently studied in different communities. Several NP-hardness proofs have appeared [6, 11, 13]. NP-hardness even holds for hypergraphs with $d = 3$ [14, 18], while $d \leq 2$ allows for

polynomial-time solvability [18] because then the input hypergraph is itself an optimal solution. If the hypergraph is closed under intersections, that is, every intersection of hyperedges is also a hyperedge, then SID also becomes polynomial-time solvable [4, Lemma 1]. Polynomial-time approximability has also been intensely studied, providing various logarithmic-factor approximation algorithms [1, 6, 18] and inapproximability results (implying that logarithmic-factor approximation algorithms are likely to be optimal) [1, 18]. The currently best exact algorithm for SID has a running time of $O(n^{2k}/4^k + n^2)$ [18]. In addition, in a series of papers it has been shown that SID can be solved in polynomial time if $2 \leq m \leq 4$ [9, 28, 29].

Much research focused on finding solutions of special structure [2, 3, 4, 7, 19, 20]. For example, it has been shown that it is possible to test in linear time whether there is a solution that is a tree, a path, or a cycle [4, 19]. Conversely, it is NP-hard to test for planar solutions [19] or 2-outerplanar solutions [4]. 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 [22]; this variant can be solved in $O(n^2(m + \log(n)))$ time [21]. Finally, in the context of overlay networks, it is of specific interest to construct solutions with small maximum and small average vertex degree [6, 26]; the latter is achieved by SID.

Our Contribution. We start by revealing an incorrectness in two (very similar) plausible data reduction rules used in previous work [14, 18] by constructing a counterexample that shows their failure (Section 3). Based on this, we provide both refined and completely new data reduction rules and prove their correctness and effectiveness. Many of our data reduction rules also work in a parameter-independent fashion, that is, they do not require knowledge of the parameter value which is just used for the mathematical analysis.

Generalizing previous work [9, 28, 29], we then show that SID can be solved in linear time if the input hypergraph contains only a constant number m of hyperedges (Section 4). This implies that SID is fixed-parameter tractable with respect to the parameter m.

For hyperedges of size $d \leq 4$ we show that SID can be reduced in polynomial time to an equivalent instance of $O(\phi)$ vertices[1], known as "polynomial-size problem kernel" in parameterized algorithmics (Section 5.1). Making use of the developed data reduction rules, we further show that, for arbitrary d, SID can be solved in $d^{18d\phi} \cdot \text{poly}(|H|)$ time (Section 5.2). Here $|H|$ denotes $\sum_{F \in \mathcal{F}} |F|$. Our result implies that SID with small hyperedges becomes tractable if the solution is required to be almost a tree, generalizing the polynomial-time algorithm for trees [19].

2. Preliminaries.

Graphs and Hypergraphs. Given an undirected graph $G = (V, E)$ with vertex set V and edge set E, we use $E(G)$ to denote the edge set E of G. We denote by $G[V']$ the *subgraph* $(V', \{e \subseteq V' \mid e \in E\})$ *of* G *induced* by V'. We also use $G - V'$ as a shorthand for $G[V \setminus V']$. A *feedback edge set* of a graph G is a set of edges whose removal makes G a forest. The *feedback edge number* ϕ of G is the size of any minimum feedback edge set. If G is connected, then the feedback edge number is $|E| - |V| + 1$.

Let V be a finite set and let $\mathcal{F}$ 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

[1] Recall that ϕ denotes the feedback edge number of an optimal solution G, that is, the minimum number of edges whose removal makes G acyclic.

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 in linear time. We use n to denote $|V|$ and $|H|$ to denote $\sum_{F\in\mathcal{F}} |F|$. 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, that is, $\mathcal{F}(v) := \{F \in \mathcal{F} \mid v \in F\}$. If $u, v \in V$ and $\mathcal{F}(u) \supseteq \mathcal{F}(v)$, 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 $\mathcal{F}' = \{F \subseteq V' \mid F \in \mathcal{F}\}$. By *removing* a vertex v from H, we mean taking the hypergraph $H' = (V \setminus \{v\}, \mathcal{F}')$, where $\mathcal{F}'$ is obtained from $\{F\setminus\{v\} \mid F \in \mathcal{F}\})$ by removing the empty set and singleton sets. A *hyperwalk* between vertices u and v is an alternating sequence of vertices and hyperedges starting in u and ending in v such that succeeding elements are incident. A hypergraph is *connected* if there is a hyperwalk between every pair of vertices.

The *covering graph* of hypergraph $H = (V, \mathcal{F})$ is the directed graph $G_C = (V, \{(u, v) \mid \mathcal{F}(u) \supseteq \mathcal{F}(v)\})$. In other words, G_C contains an arc (u, v) if and only if u covers v. Note that G_C is transitive. Some of our reduction rules construct the covering graph of H as a subroutine. The following lemma bounds the running time for this step.

LEMMA 2.1. *Given a hypergraph $H = (V, \mathcal{F})$ one can construct the covering graph G_C in $O(n \cdot |H|)$ time.*

Proof. Initialize G_C as $(V, V \times V)$. Then, for each $F \in \mathcal{F}$, remove the arcs in $(V \setminus F) \times F$ from G_C in $O(n \cdot |F|)$ time. Clearly, if (u, v) is an arc of the resulting directed graph G_C, then there is no hyperedge containing v but not u or, equivalently, $\mathcal{F}(u) \supseteq \mathcal{F}(v)$. If G_C does not contain the arc (u, v), then, by the construction of G_C, there is a hyperedge $F \in \mathcal{F}$ such that $v \in F$ but $u \notin F$. Thus, G_C contains exactly the edges (u, v) such that u covers v. ∎

Parameterized Complexity. The concept of parameterized complexity was pioneered by Downey and Fellows [8] (see also the textbooks [15, 25]). 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 is a nonnegative integer. A parameterized problem L is *fixed-parameter tractable* if there is an algorithm that decides whether $(x, k) \in L$ in $f(k) \cdot |x|^{O(1)}$ time, where f 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* [16]. Here, the goal is to transform a given problem instance (x, k) in polynomial time into an equivalent instance (x', k') with parameter $k' \leq f(k)$ such that the size of (x', k') is upper-bounded by $g(k)$, where g and f are functions only depending on k. If this is the case, we call the instance (x', k') a (problem) *kernel* of size $g(k)$.

Data reduction is usually presented as a series of *reduction rules*. These are polynomial-time algorithms that take as input an instance of some decision problem and produce another instance of the same problem 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. We call an instance I of a parameterized problem *reduced* with respect to a reduction rule if the reduction rule does not apply to I. That is, carrying out the reduction rule yields an unchanged instance.

3. Fundamental Observations on Twins in Optimal Solutions. In this section, we show that a previously proposed data reduction rule for SID is incorrect. We also show some properties of SID's optimal solutions and some data reduction rules that are used in our algorithms.

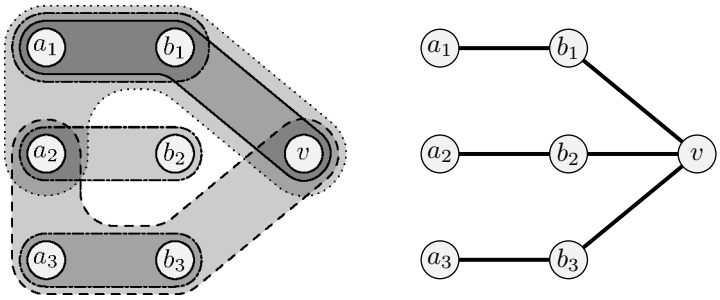

Figure 2: An example showing parts of the hypergraph H' in the proof of Lemma 3.1, herein $x = 3$. Left: some hyperedges from H'. Right: the $2x$ edges that can be assumed to be in an optimal solution for H'.

A very natural approach to identify edges of optimal solutions is to look for vertices u and v such that v covers u, that is, $\mathcal{F}(v) \supseteq \mathcal{F}(u)$. 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.*

Proof. Let $F \in \mathcal{F}$ be a hyperedge with $u \in \mathcal{F}$. Since $G[F]$ is connected, u has degree at least one in $G[F]$. Since v is the only neighbor of u in G it follows that $v \in F$. □

It is thus tempting to devise a data 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 [14], or where one covers the other [18]. The variant of these reduction rules that applies less often reads as follows.

Rule 1. *If hypergraph $H = (V, \mathcal{F})$ contains twins u and v, 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 3.1. *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 $x \geq 3$ be an arbitrary integer. Consider the hypergraph $H = (V, \mathcal{F})$, with vertex set $V = \{u, v, a_1, \ldots, a_x, b_1, \ldots, b_x\}$ and hyperedge set $\mathcal{F}$ which is the union of the following sets of hyperedges:

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

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

Now, let $(H' = (V', \mathcal{F}'), 3x - 1)$ be an instance that results from $(H, 3x)$ by applying Rule 1 to u and v, that is, removing u from H and decreasing the solution

size bound k by one. An example showing parts of the hypergraph H' for $x = 3$ is given in Figure 2. Then, $V' = V \setminus \{u\}$ and $\mathcal{F}'$ consists of the following hyperedges:

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

We show that every solution for H' has at least $3x$ edges and, thus, that $(H', 3x-1)$ is a no-instance. First, every solution for H' contains the x edges corresponding to the size-two hyperedges of $\mathcal{F}_1$. Furthermore, due to the hyperedges in $\mathcal{F}'_2$, for each $i \in \{1, \ldots, x\}$, every solution contains $\{v, a_i\}$ or $\{v, b_i\}$. By the symmetry between a_i and b_i in the hypergraph H', assume without loss of generality that an optimal solution contains the edge $\{v, b_i\}$ for all $i \in \{1, \ldots, x\}$ (the set of these edges plus $\mathcal{F}_1$ is shown in Figure 2). Now, let $G' = (V', E')$ be such a solution for H'. 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 set of the remaining a_is. If $A_1 = \emptyset$, then G' contains at least $3x$ edges. We show that if $A_1 \neq \emptyset$, then the graph G' also has at least $3x$ edges. Assume that G' is optimal and that every optimal solution has at least $y > 0$ vertices in A_1. For every hyperedge $F = \{v, a_i, b_i, a_j\}$ with $a_j \in A_1$ and $i \in \{1, \ldots, x\} \setminus \{j\}$, 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 $\{v, a_j\}$: Since $j \neq i$, there is only one hyperedge, namely F, that contains both b_i and a_j. Clearly, $G'[F]$ can also be made connected by adding $\{v, a_j\}$ instead. This implies an optimal solution with $y-1$ vertices in A_1, contradicting our choice of y. Hence, G' contains no edges $\{b_i, a_j\}$ with $i \neq j$. Consequently, in order to make each hyperedge $\{v, a_i, b_i, a_j\} \in \mathcal{F}'_3$ with $a_j \in A_1$ connected, there is an edge between a_i and a_j.

Hence, G' has $y \cdot (x - y)$ edges between A_1 and A_2, $\binom{y}{2}$ edges between vertices in A_1 and $x-y$ further edges between v and A_2. Altogether the total number of edges in G' is thus at least

$$2x + y \cdot (x - y) + \binom{y}{2} + x - y = 3x + \frac{y \cdot (2x - y - 3)}{2}$$
$$\geq 3x + \frac{y \cdot (x - 3)}{2} \geq 3x.$$

This implies that $(H', 3x - 1)$ is a no-instance. □

As one can see from the proof, the main reason for the incorrectness of Rule 1 is the incorrect assumption that there is an optimal solution which adds an edge between twins. However, 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 all other edges incident with u to v.

LEMMA 3.2. *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.*

Proof. Assume that u has degree at least two in G, that is, u has some neighbor $w \neq v$. Then, obtain a modified graph G' by replacing the edge $\{u, w\}$ by the edge $\{v, w\}$. The modified graph G' has the same number of edges as G. Furthermore, the two endpoints of the removed edge $\{u, w\}$ are still connected in each hyperedge

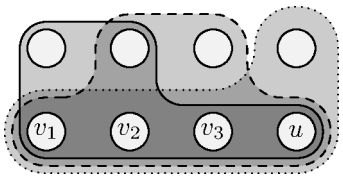

Figure 3: A hypergraph to which Rule 3 applies.

that contains them: Since v covers u, such a hyperedge also contains v which is in G' a common neighbor of u and w. Hence, G' is also an optimal solution. □

The above lemma immediately implies the following reduction rule.

RULE 2. *If hypergraph $H = (V, \mathcal{F})$ contains vertices u, v such that v covers u and if there is an optimal solution G containing the edge $\{u, v\}$, then remove u from H and decrease k by one.*

Of course, it is not clear how to efficiently determine whether such an optimal solution exists. We later exhibit hypergraph structures that enforce this and then use Rule 2 as a subroutine.

LEMMA 3.3. *Rule 2 is correct.*

Proof. If there is a solution of size at most k for H, then the condition of the rule together with Lemma 3.2 imply that there is a solution G of size at most k in which u has degree one and is adjacent to v. Then, $G - \{u\}$ is a solution of size at most $k-1$ for H with u removed. Conversely, if there is a solution G' of size at most $k-1$ for H with u removed then adding the edge $\{u, v\}$ gives a solution of size at most k for H since all hyperedges F containing u also contain v and the corresponding subhyperedges $F \setminus \{u\}$ induce connected subgraphs of G'. □

Note that the correctness of Rule 2 together with Lemma 3.1 implies that there are instances in which a twin class is an independent set 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 each hyperedge containing u also contains at most *two* vertices that do not cover u, then Rule 1 is correct.

RULE 3. *If there are $q+1$ vertices u and $v_1, \ldots, v_q$ in the hypergraph $H = (V, \mathcal{F})$, q being a positive integer, such that each v_i covers u and if for each hyperedge $F \in \mathcal{F}(u)$, $|F| \leq q + 3$ holds, then remove u from H and decrease k by one.*

See Figure 3 for a hypergraph to which Rule 3 applies with $q = 3$. Another example is the hypergraph H defined in Lemma 3.1 without the hyperedges in $\mathcal{F}_3 \cup \mathcal{F}_4$. Then $q = 1$ and all hyperedges are of size at most four.

LEMMA 3.4. *Rule 3 is correct and can be applied exhaustively in $O(n \cdot |H|)$ time.*

Proof. Let $H = (V, \mathcal{F})$ be a hypergraph, Q be the vertex set $\{v_1, \ldots, v_q\} \subseteq V$, and u be a vertex in V such that Rule 3 applies. Further, let N denote the set of neighbors of u in an optimal solution G. If $N \cap Q \neq \emptyset$, then the correctness follows from the correctness of Rule 2. Hence, we assume that $N \cap Q = \emptyset$.

Case 1: N contains a neighbor w of some $v \in Q$ in G. Let G' be the result of removing the edge $\{u, w\}$ from G and adding $\{u, v\}$. If there is some hyperedge F such that $G'[F]$ is disconnected, then $u, w \in F$. Since v covers u, also $v \in F$. Then, there is a path between u and w via v in $G'[F]$, implying that $G'[F]$ is connected.

Case 2: N contains no neighbor of any $v \in Q$. Then, for each $F \in \mathcal{F}(u)$,

$|F \cap N| \leq 1$ since otherwise, $|F| \leq q+3$ implies that $F = \{u\} \cup Q \cup (F \cap N)$ and then $G[F]$ does not contain a path between u and any vertex in Q. Thus, $G' := G - \{u\}$ is a solution for the hypergraph H' that results from H by removing u. Note that G' has at least one edge less than G. Since all hyperedges of H that contain u are supersets of Q, adding u with the edge $\{v_1, u\}$ to G' results in a solution for H, which is optimal since $|E(G')| + 1 \leq |E(G)|$.

It remains to prove the running time bound. For a vertex u, let Q be the set of vertices covering u. It is not hard to see that if Rule 3 applies to any subset of Q, then it also applies to Q. Reversing all arcs in the covering graph G_C of H allows us to compute $|Q|$ in constant time per vertex. Thus, assuming that the size of a hyperedge can be computed in constant time, Rule 3 can be applied exhaustively to $H = (V, \mathcal{F})$ in $O(n \cdot |H| + n \cdot \sum_{u \in V} |\mathcal{F}(u)|) = O(n \cdot |H|)$ time. □

As a corollary of Lemma 3.4, we also obtain correctness of the following rule since it is a special case of Rule 3. This rule will be useful in Section 5.

RULE 4. *If there are two vertices u and v such that v covers u 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, then Rule 4 is not correct as shown by Lemma 3.1.

4. 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 [18, Theorem 8] relied on Rule 1 and is therefore incorrect.[2] The intuition behind Rule 1 is that it is optimal to connect a twin class by a spanning tree and to subsequently represent this twin class by a single vertex. This approach results in an instance with at most 2^m vertices which would imply fixed-parameter tractability. As the counterexample in Lemma 3.1 shows, a twin class can be disconnected in the subgraph induced by every optimal solution. Thus, in order to restore the fixed-parameter tractability 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 4.1. *Every input instance of* SID *with $k \geq \binom{2^m}{2} + n - 1$ is a yes-instance.*

Proof. We show how to construct a solution G with less than $\binom{2^m}{2} + n$ edges. The main idea is to first partition the vertex set into two subsets V' and $V \setminus V'$. We then construct a solution G' for the hypergraph H' resulting from H by removing the vertices in $V \setminus V'$. Finally, we obtain a solution for H by adding each time one edge connecting each vertex in $V \setminus V'$ to the graph G'.

Recall that a twin class is a maximal set of vertices that mutually cover each other. Let V' consist of exactly one vertex from each twin class of H. Clearly, $|V'| \leq 2^m$ where m is the number of hyperedges. Let H' be the hypergraph resulting from removing all vertices that are not in V' from H. Obviously, a complete graph G' for V' is a solution for hypergraph H'. This solution has $\binom{2^m}{2}$ edges.

We now extend G' to a solution G for H as follows. Add each vertex $v \in V \setminus V'$ to G'. Furthermore, add an edge incident to v and its twin in V' (recall that at least one vertex of each twin class is in V'). Let G be the resulting graph. Since there is at least one twin class, $|V \setminus V'| \leq n - 1$ and hence G contains at most $\binom{2^m}{2} + n - 1$ edges.

[2] The theorem of Hosoda et al. [18] actually states that an optimal solution can be computed in polynomial time if $m \leq f(n)$ where f is some specific function. This is equivalent to fixed-parameter tractability: if $m \leq f(n)$, then one can apply the polynomial-time algorithm, otherwise $m > f(n)$ implies that the instance size depends only on m.

Graph G is a solution for H since for each hyperedge $F \in \mathcal{F}$, the subgraph $G[F]$ is connected: $G[V' \cap F]$ is a complete graph and each vertex in $F \setminus V'$ is adjacent to its twin in $F \cap V'$. Thus, $G[F]$ is connected. □

The upper bound provided by Lemma 4.1 grows exponentially in the number of hyperedges. It would be interesting to replace this exponential dependence by a polynomial function. However, this is not possible in general; there are instances that require a solution with at least $2^{\Omega(m)} + n$ edges, see Appendix A.

Lemma 4.1 directly yields the correctness of the following reduction rule.

RULE 5. *If $k \geq \binom{2^m}{2} + n - 1$, then answer "yes".*

Our aim is now to shrink the size each of the 2^m twin classes to be bounded by a function of m. For this, the following rule removes vertices from large twin classes.

RULE 6. *Let (H, k) be an instance that is reduced with respect to Rule 5. 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 6, we show that there is a solution G that has the following property concerning its low-degree vertices. In the following, by degree-ℓ vertices we refer to vertices of degree exactly ℓ.

LEMMA 4.2. *Let $H = (V, \mathcal{F})$ be a connected hypergraph and $|V| \geq 3$. Then, there is a solution $G = (V, E)$ such that for each twin class T of H, graph G has*

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

Proof. Let G be a solution for H. We show how to transform G into a solution G^* which fulfills both properties of the lemma. First, suppose that there is a nonempty set $\mathcal{T} = \{T_1, \ldots, T_q\}$ of twin classes of H such that at least two vertices of each T_i have degree-one neighbors in G. Now, modify G as follows. Pick an arbitrary vertex t_i from each T_i such that t_i has a degree-one neighbor and label t_i as the *one* vertex in twin class T_i that will have degree-one neighbors in the modified solution G^*.

Then, as long as $\mathcal{T}$ is nonempty, pick an arbitrary T_i and an arbitrary vertex $t' \in T_i \setminus \{t_i\}$. Let X be the set of degree-one neighbors of t' in G. Note that $t_i \notin X$ as, otherwise, t_i and its neighbor would form a connected component of size two in G. This contradicts the fact that G is a solution because by definition H is connected and contains at least three vertices. We remove all edges between X and t' and make all vertices in X adjacent to t_i. Let G' be the resulting graph. Observe that G' has the same number of edges as G. Moreover, G' is also a solution: The vertices of X were not part of any shortest path between two vertices in $V \setminus X$. Hence, $G[V \setminus X]$ is a solution for the hypergraph obtained from H by removing each vertex in X. In particular, for each $F \in \mathcal{F}$ containing t_i, vertex t_i is connected to all other vertices in $G[F \setminus X]$. Therefore, adding each vertex in X as a degree-one neighbor to t_i (which results in the graph G') produces a solution for H.

The above shows the correctness of the first modification step. After this step we update the set $\mathcal{T}$ by performing the modification step, again for some arbitrary twin class of H that contains at least two vertices with degree-one neighbors in G'. We repeat this process until $\mathcal{T}$ is empty. In order to show that we can obtain a solution G^* in which $\mathcal{T}$ is empty, we need to show that we only have to perform a finite number of modification steps. To this end, note that in each step the degree of the labeled vertex $t_i \in T_i$ increases and only the degree of t' decreases. Since t' is a twin of t_i, the degree of all other labeled vertices stays the same. Hence the number of modification steps is finite. Summarizing, there is a solution that fulfills Property 1 of the lemma.

We now show that such a solution can be further modified such that it also fulfills

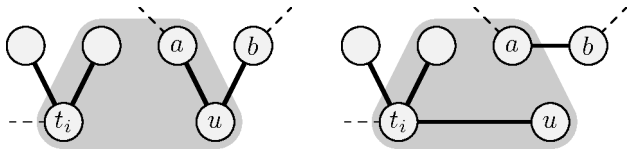

Figure 4: A possible situation before (left) and after the second modification step (right) in the proof of Lemma 4.2. The gray region represents a twin class.

Property 2. Let G be a solution that fulfills Property 1 and let $\mathcal{T} = \{T_1, \ldots, T_q\}$ denote the twin classes for which there are at least two degree-two vertices in G. For each T_i do the following. Since the first property is satisfied, T_i contains at least one degree-two vertex u such that its two neighbors a and b have degree at least two, respectively. Then, remove the two edges incident with u, make a and b adjacent, and make u adjacent to either the one vertex in T_i that has degree-one neighbors, or, if such a vertex does not exist, another degree-two vertex in T_i. See Figure 4 for an illustration of this modification step. Note that the neighbors of t_i and u do not have to be contained in T_i. Nevertheless, for each hyperedge F the connected component of a and b is connected to the one of t_i (if it exists) in $G[F]$, which we now exploit to show the correctness of the modification step.

Let G' be the modified graph and let v denote the new neighbor of u in G'. Clearly, G' has the same number of edges as G. To show that G' is also a solution, we consider each $F \in \mathcal{F}$ that contains u and at least one of a and b and show that $G'[F]$ is connected.

Case 1: Exactly one of a and b is in F. Then, u has degree one in $G[F]$, and thus $G[F] - \{u\}$ is connected. In particular, the twin v of u has a path to all vertices in $G[F] - \{u\}$. Hence, "reinserting" u and making it adjacent to v results in a connected graph isomorphic to $G'[F]$.

Case 2: Both a and b are in F. Since $G[F]$ is connected, the graph G'' that is obtained from $G[F]$ by removing u and making a and b adjacent is also connected. Again, this means that the twin v of u (in T_i) has in G'' a path to all other vertices. As above, "reinserting" u and making it adjacent to v yields a connected graph isomorphic to $G'[F]$.

Note that since neither a nor b are degree-one vertices and by the choice of v, the above modification does not result in a solution in which a twin class has more than one vertex that is a neighbor of degree-one vertices. Consequently, the modification can be applied to all T_i's without losing the first property. Hence, there is a solution fulfilling both properties. □

With the above lemma at hand, we can show the correctness of Rule 6. The proof idea is as follows. Using Lemma 4.2, we show that only $O(2^m)$ vertices of every twin class T have degree at least three but at least one low-degree neighbor. Consequently, for sufficiently large $|T|$ we can assume that one vertex of T has degree one in G: Otherwise, there are many degree-(≥ 3) vertices in G that have only degree-(≥ 3) neighbors in G. This, however, pushes the number of edges in G above the guarantee given by Rule 5.

Lemma 4.3. *Rule 6 is correct.*

Proof. Let (H, k) denote the original instance and $(H', k-1)$ an instance resulting

from one application of Rule 6. We show that both instances are equivalent. It is easy to see that if $(H', k-1)$ is a yes-instance, then (H, k) is also a yes-instance: Let G' be the solution for H'. Pick one vertex $u \in T \setminus \{v\}$, make v adjacent to u, and call the resulting graph G. Then, G has one more edge than G'. Moreover, G is also a solution since for each $F \in \mathcal{F}$ containing v, the subgraph $G'[F \setminus \{v\}]$ is connected. This implies that $G[F]$ is also connected (since $F \setminus \{v\}$ contains u).

For the other direction of the equivalence, suppose that G is a solution for H. Since H is reduced with respect to Rule 5, graph G has $k < \binom{2^m}{2} + n$ edges. Let twin class T and vertex v be as described in Rule 6. To show that $(H', k-1)$ is also a yes-instance, we show that, since $|T| > 4^m + 7 \cdot 2^m + 1$, hypergraph H has also a solution with k edges where vertex v has degree one. Herein, we assume that H and thus also G are connected. The case for disconnected input hypergraphs follows easily.

First, we show that there are at most $4^m + 7 \cdot 2^m$ vertices whose degree in G is at least three. Obviously, this upper-bounds the number of vertices with degree at least three in the twin class T as well. We consider two subsets of this vertex set: by X we denote the vertices of degree at least three that have only neighbors of degree at least three in G, and by Y we denote the other vertices of degree at least three. Note that Lemma 4.2 implies that Y has at most $(1+2) \cdot 2^m = 3 \cdot 2^m$ vertices: First, the number of twin classes is at most 2^m. Second, in G each twin class has at most one vertex that has degree-one neighbors. Finally, in G each twin class has at most one degree-two vertex. Hence, there are at most $2 \cdot 2^m$ neighbors of degree-two vertices.

It remains to upper-bound the size of X. We do this by deriving a lower bound on the number of edges in G and then show that, for large X, this lower bound exceeds k, contradicting the fact that (H, k) is a yes-instance. To this end, let $Z = V \setminus (X \cup Y)$ denote the set of vertices that have degree one or two in G. We partition the edges of G into two subsets $E_{X \cup Y}$, which contains edges with both endpoints on vertices of degree at least three, and $E_{Y \cup Z}$ which contains all other edges. Since we assume that G is connected, $|E_{Y \cup Z}| \geq |Z| - 1$. The number of edges in $E_{X \cup Y}$ is at least $3|X|/2$ since all vertices in X have degree at least three and only neighbors in $X \cup Y$. If $|X| \geq 2 \cdot \left(\binom{2^m}{2} + 3 \cdot 2^m\right)$, then

$$|E_{X \cup Y}| \geq \frac{3|X|}{2} = |X| + \frac{|X|}{2} \geq |X| + |Y| + \binom{2^m}{2},$$

where the last inequality holds because $|Y| \leq 3 \cdot 2^m$. Hence, the number of edges in G is

$$|E_{X \cup Y}| + |E_{Y \cup Z}| \geq |X| + |Y| + \binom{2^m}{2} + |Z| - 1 = n + \binom{2^m}{2} - 1.$$

This contradicts the assumption that (H, k) is a yes-instance, since $k < n + \binom{2^m}{2} - 1$ after application of Rule 5. Hence, we have $|X| < 2 \cdot (\binom{2^m}{2} + 3 \cdot 2^m) = 4^m + 5 \cdot 2^m$.

Now we can upper-bound the number of vertices in the twin class T that have degree at least two. In addition to the vertices of X, class T can contain at most one vertex that is a neighbor of degree-one vertices, at most $2 \cdot 2^m$ vertices that are neighbors of degree-two vertices, and at most one degree-two vertex. Therefore, if T contains more than $4^m + 7 \cdot 2^m + 1$ vertices, then at least one of them is a degree-one vertex in G. Without loss of generality, we can assume that this is v. □

It now only remains to combine the above results to obtain a problem kernel for SID parameterized by the number m of hyperedges. Moreover, this kernel can be computed in linear time, thus yielding linear-time fixed-parameter tractability.

THEOREM 4.4. *An instance of* SUBSET INTERCONNECTION DESIGN *can be reduced to an equivalent one of size at most* $O(8^m \cdot m)$ *in* $O(|H|)$ *time.*

Proof. The kernelization algorithm first applies Rule 5 and then exhaustively applies Rule 6. After this, the size of each twin class in H is at most $O(4^m)$. Hence, the instance has size $O(8^m \cdot m)$: The input hypergraph H has at most 2^m twin classes, each containing $O(4^m)$ vertices. Therefore, the total number n of vertices is $O(8^m)$. The overall instance size follows.

The running time of the kernelization algorithm can be upper-bounded as follows. Clearly, Rule 5 runs in $O(|H|)$ time. In order to apply Rule 6, we first compute a partition of V into the twin classes. This can be done as follows. We start with one set containing V and then consider an arbitrary hyperedge $F \in \mathcal{F}$. The vertices that are in F are in a different twin class than the vertices that are not in F. Hence, using F we partition V into two subsets. We repeat the partitioning for all hyperedges, each time using the current hyperedge to update the partition. The partitioning can be done in $O(|F|)$ time [17, Lemma 1]. This process thus takes $O(|H|)$ time. Its output is a partition of V into all different twin classes. For each twin class, we check whether Rule 6 can be applied. Instead of applying the rule right away, we label the vertices of the twin classes that will be removed and decrease k by the overall number of labeled vertices. After all twin classes have been processed, we remove all labeled vertices from each $F \in \mathcal{F}$, again in linear time. Thus, the overall running time is $O(|H|)$. ∎

COROLLARY 4.5. SUBSET INTERCONNECTION DESIGN *can be solved in* $2^{O(m8^m)} + O(|H|)$ *time.*

Proof. By Theorem 4.4 we can obtain in $O(|H|)$ time an equivalent instance with at most $c8^m$ vertices for some constant c (c can be selected as 2 for $m \geq 3$). Hence, by Theorem 4.1, we can assume that $k < \binom{2^m}{2} + n - 1 \leq 4^m + c8^m \leq (c+1)8^m$. Now we try all subsets of k edges out of the at most $\binom{c8^m}{2} \leq c^2 8^{2m}$ possible edges on at most $c8^m$ vertices. For each of them we test, whether it is a solution in $O(mk) = O(m8^m)$ time. Since there are at most $\binom{c^2 8^{2m}}{k} \leq \binom{c^2 8^{2m}}{(c+1)8^m} \leq (c^2 8^{2m})^{(c+1)8^m} = 2^{O(m8^m)}$ such sets, we get the $2^{O(m8^m)}$ bound for the running time of this part of the algorithm. ∎

5. Data Reduction Rules for Sparse Solutions. In this section, we present a set of reduction rules that identify and remove parts of the instance which either produce tree-like induced subgraphs or long induced paths in the solution. We analyze the power of these data reduction rules by showing that, for maximum hyperedge size $d \leq 4$, they yield a problem kernel for the parameter feedback edge number ϕ and that, for general d, they can be used to obtain fixed-parameter tractability for the combined parameter (d, ϕ). The reason for obtaining a weaker result for $d > 4$ is that Rule 1 is indeed correct if $d \leq 4$ but not in general (see Rule 4). Applying this rule removes a certain class of vertices from the input graph that seem to be hard to identify for $d > 4$.

Although k appears in our data reduction rules, they are applicable regardless of the value of k. Hence, the data reduction rules can be applied also to the optimization version of SID.

5.1. A Problem Kernel for the Parameter Feedback Edge Number ϕ for $d \leq 4$. We now describe how we can remove all but $O(\phi)$ vertices from a SID

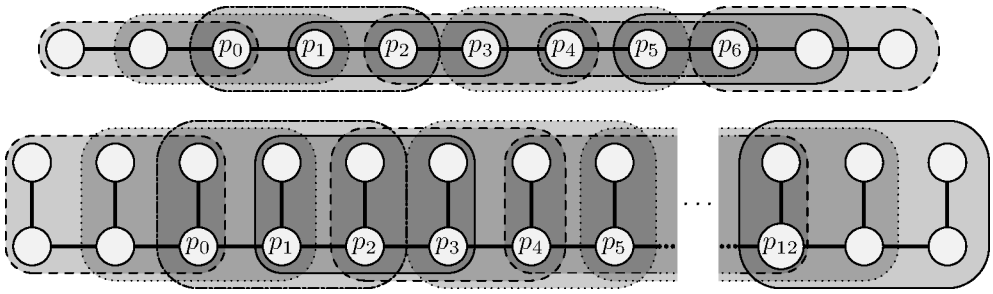

Figure 5: Two hypergraphs to which Rule 7 applies along with optimal solutions.

instance with $d \leq 4$ in $O(n \cdot m^3)$ time by using Rule 4 (Section 3) and an additional reduction rule (Rule 7 below). Informally, the parameter ϕ upper-bounds the number of vertices that are in cycles and have degree at least three. Using Rule 4 we can remove vertices that have degree one in solutions. Hence, to reduce the overall number of vertices to $O(\phi)$, we also have to remove vertices that are in long induced paths in the solution. This is the purpose of Rule 7. This rule is also needed in Section 5.2 which deals with the case $d \geq 5$. Therefore, we formulate the rule in a more general way than needed for the special case $d \leq 4$.

RULE 7. *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 $\mathcal{F}_P := \bigcup_{p \in P} \mathcal{F}(p)$ such that*

1. *no $p_i \in P$ covers any $p_j \in P$ with $j \neq i$,*
2. *for each $F \in \mathcal{F}_P$ we have $F \cap P = \{p_i, \ldots, p_j\}$ for some $0 \leq i \leq j \leq 2d$,*
3. *for each $F \in \mathcal{F}_P$ 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 $F \in \mathcal{F}_P$ such that $F \cap P = \{p_i\}$ for any $0 < i < 2d$,*

then do the following.

For every $F \in \mathcal{F}_P$ 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 plus the fact that the hyperedges have size at most d enforce that there is a solution G for such a hypergraph that makes $G[P]$ a long induced path with endpoints p_0 and p_{2d}. Below, we use $P_I := \{p_1, \ldots, p_{2d-1}\}$ to denote the set of inner vertices of this path structure. Condition 3 ensures that in the solution in which $G[P]$ is a path all vertices that are in hyperedges with only vertices of P_I can be made degree-one neighbors of some vertex of P_I.

Two examples of hypergraphs to which Rule 7 applies are shown in Figure 5 with values $d = 3$ and $d = 6$, respectively.

In order to prove the correctness and running time of the rule, we first make some observations on the structure of the subhypergraph that consists of the hyperedge set $\mathcal{F}_P$. The first observation is about the structure of the hyperedges along the presumed path containing P.

OBSERVATION 2. *Let H be a hypergraph and $P \subseteq V$ satisfying the conditions of Rule 7. For every $p_i \in P_I$ 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 $F_{2d}^+ \cap P = \{p_{2d}\}$.*

Proof. Consider some $i \in \{1, \ldots, 2d-1\}$. By Condition 1 of Rule 7, there is some hyperedge containing p_i but not p_{i-1}. Now, by Condition 4, this hyperedge contains at least one further vertex from P, and by Condition 2 it thus contains p_{i+1}. Hence, there is a hyperedge F_i^+ containing p_i and p_{i+1} that does not contain p_{i-1}. The existence of F_i^- follows from a symmetric argument. Finally, Condition 1 implies the existence of F_0^- and F_{2d}^+. ∎

For the second observation we consider the hyperedges that only intersect with the set of inner vertices P_I. To this end, let $\mathcal{F}_I = \{F \in \mathcal{F}_P \mid F \cap P \subseteq P_I\}$ be the hyperedges of $\mathcal{F}_P$ that contain neither p_0 nor p_{2d} and let $W := \bigcup_{F \in \mathcal{F}_I} F$ be the union of all vertices that are incident with some hyperedge in $\mathcal{F}_I$. Due to Observation 2, $\mathcal{F}_I$ is not empty and $W \cap P = P \setminus \{p_0, p_{2d}\}$. Each vertex v in $W \setminus P$ is covered by some vertex of $P \setminus \{p_0, p_{2d}\}$: By the definition of W, there is a hyperedge $F \in \mathcal{F}_I$ with $F \cap \{p_0, p_{2d}\} = \emptyset$ that is incident with v. By Condition 3, v is covered by some $p_i \in P$. Since $F \cap \{p_0, p_{2d}\} = \emptyset$, we have $p_i \neq p_0$ and $p_i \neq p_{2d}$. Altogether, W thus has the following property.

Observation 3. *For each $v \in W$ either $v \in P_I$ or there is a $p_i \in P_I$ such that p_i covers v.*

Informally, the above two observations imply that the subhypergraph $H[W]$ has a solution in which P_I is an induced path and all other vertices of W have degree one. The final observation is used to show that this solution for $H[W]$ is also optimal. It is based on the fact that any solution for a connected input hypergraph is a connected graph.

Observation 4. *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 now show the correctness of the rule.

Lemma 5.1. *Rule 7 is correct.*

Proof. Let H, P and $\mathcal{F}_P$ be as described in Rule 7 and, as above, let $P_I = \{p_1, \ldots, p_{2d-1}\}$, $\mathcal{F}_I = \{F \in \mathcal{F}_P \mid F \cap P \subseteq P_I\}$ and $W = \bigcup_{F \in \mathcal{F}_I} F$. Consider an arbitrary optimal solution G' and let G be the graph obtained from G' by removing all edges in $G'[W]$ and then adding the edges $\{p_i, p_{i+1}\}$ for $i \in \{1, \ldots, 2d-2\}$, and, for every $v \in W \setminus P$ adding one edge $\{v, p_i\}$ where p_i covers v. We prove that G is an optimal solution.

First, we show that G is a solution, that is, $G[F]$ is connected for all $F \in \mathcal{F}$. For this, let $F \in \mathcal{F}$ and $F_W = F \cap W$. Observe that if $|F_W| < 2$, then $G[F]$ is connected. Hence, assume $|F_W| \geq 2$ and let $u, v \in F_W$ such that $\{u, v\} \in E(G')$. We show that u and v are connected in $G[F_W]$. This directly implies that $G[F]$ is connected, since any path in $G'[F]$ using $\{u, v\}$ can then use the path between u and v in $G[F_W]$ instead. By Observation 3, we have $F_W \cap P \supseteq \{p_i, p_j\}$ for some $1 \leq i \leq j \leq 2d-1$, and Condition 2 yields $F_W \cap P \supseteq \{p_i, p_{i+1}, \ldots, p_j\}$. Without loss of generality, we may assume that either $u = p_i$ or $\{u, p_i\} \in E(G)$. Similarly, we may assume that either $v = p_j$ or $\{v, p_j\} \in E(G)$. Hence, it suffices to show that $G[\{p_i, \ldots, p_j\}]$ is connected. This is directly implied by the construction of G. Thus, there is a path between u and v and G is a solution.

We now prove the optimality of G. For this, we first show that the induced subhypergraph $H[W] = (W, \mathcal{F}_I)$ is connected, which implies a lower bound on the number of edges in $G'[W]$. Let $u, v \in W$. We construct a hyperwalk between u and v using only hyperedges of $\mathcal{F}_I$. By Observation 3, it suffices to show that there is a hyperwalk between any p_i, p_{i+1}, $1 \leq i \leq 2d-2$. Assume without loss of generality

that $i \leq d$. By Observation 2 there is some hyperedge $F \in \mathcal{F}$ with $p_i, p_{i+1} \in F$ and $p_{i-1} \notin F$ for each $i \in \{1, \ldots, 2d-1\}$. Condition 2 states that $F \cap P = \{p_i, \ldots, p_j\}$ and since $|F| \leq d$ we have $j \leq i + d - 1 < 2d$. Hence, $F \in \mathcal{F}_I$ and $H[W]$ is connected.

The fact that hypergraph $H[W]$ is connected and Observation 4 imply that $G'[W]$ contains at least $|W|-1$ edges. The graph $G[W]$ also contains $|W|-1$ edges. Since W contains all vertices p_i for $1 \leq i \leq 2d-1$, by the construction of graph G', graphs G' and G share all edges not contained in $G[W]$. Hence, G is optimal.

We have thus shown that there is an optimal solution G where each vertex $v \in W \setminus P$ is adjacent to a vertex in P that covers v. Combining this with Rule 2 we conclude that removing any vertex in $W \setminus P$ from the hypergraph and decreasing k by one results in an equivalent instance. Hence, in the following we assume that $W \subseteq P$, and hence $W = \{p_1, \ldots, p_{2d-1}\}$. It remains to show that the remaining modifications for $p_2, \ldots, p_{2d-2}$ are correct.

Let (H, k) be the original instance and (H', k') the instance resulting from an application of Rule 7 to (H, k). Note that $k' = k - 2d + 2$ since we assume that the vertex set $W = \{p_1, \ldots, p_{2d-1}\}$ and after the application of Rule 7, only the vertices p_i with $2 \leq i \leq 2d-2$ are removed. Let P^* denote the set of the removed vertices. We prove that the instances are equivalent.

First, let G be a solution with at most k edges for H as constructed in the first part of the proof. We show that we can obtain a solution for H' with at most k' edges from G. Consider a hyperedge $F \in \mathcal{F}$ such that $G[F \setminus P^*]$ is disconnected. Clearly, $F \cap P^* \neq \emptyset$ and $F \setminus P^* \neq \emptyset$. Because of Condition 2 and $|F| \leq d$, the set $F \setminus P^*$ then contains exactly one of p_1 and p_{2d-1}. Without loss of generality, let $p_1 \in F$. Remove each edge $\{v, p_i\}$ in $G[F]$ with $i \in \{2, \ldots, d\}$ and add $\{v, p_1\}$. Let G^* be the graph obtained by iterating the replacements as long as possible. We obtain a solution for H' by taking $G^*[V \setminus P^*]$: We have that $G^*[F \setminus P^*]$ is connected because any path in $G[F]$ between two vertices in $F \setminus P^*$ that used $\{v, p_i\}$ can now use $\{v, p_1\}$ instead. Since $2d-2$ edges are incident with P^* in G^*, it follows that $G^*[V \setminus P^*]$ has at most k' edges. Hence, if (H, k) is a yes-instance, then also (H', k') is a yes-instance.

For the converse, let G' be an optimal solution for H' with at most k' edges. We claim that adding the edges $\{p_i, p_{i+1}\}$ for $i \in \{1, \ldots, 2d-2\}$ yields a solution G for H. Clearly, it suffices to consider hyperedges F that intersect P^*. Condition 2 implies that $G[F \cap P^*]$ is connected. Hence, if $F \subseteq P^*$, then $G[F]$ is connected. Otherwise, $G'[F \setminus P^*]$ is connected since G' is a solution. By Condition 2, either $p_1 \in F$ or $p_{2d-1} \in F$. Since $F \cap P^*$ is connected to p_1 or to p_{2d-1} in G, this implies that $G[F]$ is connected. Note that G contains at most k edges since it contains exactly $2d-2$ edges more than G'. Hence, if (H', k') is a yes-instance, then also (H, k) is a yes-instance and the rule is correct. □

We now show the running time of Rule 7.

LEMMA 5.2. *It is possible to apply Rule 7 or to decide that it does not apply to the hypergraph in $O(m^3 d^3)$ time.*

Proof. First, recall Observation 2. In order to find an application of Rule 7 we will consecutively find hyperedges that correspond to the definitions of F_i^- and F_i^+. The algorithm is as follows.

We start by building the covering graph of H. Then, we construct an auxiliary hypergraph $H^* = (V^*, \mathcal{F}^*)$ as follows: Start with the hypergraph H and while there is a vertex in the covering graph that has an incoming arc, remove this vertex from H. Note that V^* does not contain any pair of vertices u and v such that u covers v. The running time for constructing H^* is dominated by the $O(n \cdot |H|) = O(m^2 \cdot d^2)$ time

needed for constructing the covering graph.

Intuitively, we successively discover the vertices in $p_1, \ldots, p_{2d-1}$ from Rule 7 that are contained in the F_i^- and F_i^+, giving rise to successively new hyperedges F_i^-, F_i^+. By considering only V^*, the F_i^- and F_i^+ form a very regular structure, and when p_i is fixed, then there is little choice for p_{i+1}. Hence, we can basically guess p_1 and greedily determine the successive p_i, $i > 1$, until either sufficiently many are found, or there are no choices left. Subsequently, we make this approach more formal.

We guess, trying all possibilities, which of the sets F in $\mathcal{F}^*$ is the set $F_1^+ \cap V^*$. Let us assume that our guess was correct. We then check for vertices v having the following property:

Property (a): there is a hyperedge $F' \in \mathcal{F}^*$ such that $F \cap F' = \{v\}$.

Discard the guess of F if there are more than two or less than two vertices with Property (a). Otherwise, for both choices to denote one of these vertices by p_1, proceed as follows.

Set $i := 1$ and repeat the following as long as $F \setminus \{p_1, \ldots, p_i\} \neq \emptyset$. Check whether there are vertices v in $F \setminus \{p_1, \ldots, p_i\}$ with

Property (b): there is a hyperedge $F' \in \mathcal{F}^*$ such that F' contains v and p_i, and $F' \cap F \subseteq \{p_1, \ldots, p_i, v\}$.

If there is exactly one vertex v in $F \setminus \{p_1, \ldots, p_i\}$ with Property (b), then denote this vertex p_{i+1} and set $i := i + 1$. Otherwise discard the guess of p_1.

Now, set $i = |F|$ and repeat the following as long as $i < 2d - 1$. Check for each vertex v in $V^* \setminus \{p_1, \ldots, p_i\}$ whether it has

Property (c): there is a hyperedge $F' \in \mathcal{F}^*$ such that F' contains v, $F' \cap \{p_1, \ldots, p_i\} = \{p_i\}$ and for every $u \in F' \setminus \{v, p_i\}$, and every hyperedge $F'' \in \mathcal{F}^*$ containing u and p_i we have that F'' contains v.

If there is exactly one vertex with Property (c), then denote this vertex v as p_{i+1} and set $i := i + 1$. Otherwise discard the guess of p_1.

Finally, let $L = \bigcap\{F' \in \mathcal{F}^* \mid F' \not\subseteq \{p_1, \ldots, p_{2d-1}\} \wedge (p_1 \in F')\} \setminus \{p_1, \ldots, p_{2d-1}\}$ and similarly $R = \bigcap\{F' \in \mathcal{F}^* \mid F' \not\subseteq \{p_1, \ldots, p_{2d-1}\} \wedge (p_{2d-1} \in F')\} \setminus \{p_1, \ldots, p_{2d-1}\}$. Let p_0 be an arbitrary vertex in L and p_{2d} an arbitrary vertex in R. We now check whether $p_0, \ldots, p_{2d}$ satisfies the conditions of Rule 7 in the original input hypergraph H. We claim that if there is an application of Rule 7, then the above algorithm finds it.

Denote the vertices in P in the application of the rule by $p'_0, \ldots, p'_{2d}$. Let F_i^+ and F_i^- be as in Observation 2 and denote $F'^+_i = F_i^+ \cap V^*$ and $F'^-_i = F_i^- \cap V^*$, respectively. If $p'_0, \ldots, p'_{2d}$ satisfies the assumptions of Rule 7 and $\mathcal{F}(u) = \mathcal{F}(p'_i)$, then also $p'_0, \ldots, p'_{i-1}, u, p'_{i+1}, \ldots, p'_{2d}$ satisfies the assumptions of Rule 7 and, hence, we can assume $p'_0, \ldots, p'_{2d} \in V^*$.

Consider the branch in which $F = F'^+_1$. The sets F'^-_1 and $F'^+_{|F|}$ witness the Property (a) for p'_1 and $p'_{|F|}$, respectively, and no other vertex can satisfy Property (a) by Conditions 4 and 2. Hence, we may assume $p_1 = p'_1$. For each $i \in \{1, \ldots, |F| - 1\}$, the set F'^-_{i+1} witnesses the Property (b) for p'_{i+1} and no other vertex in F has Property (b).

Next, for $i \in \{|F|, \ldots, 2d - 2\}$, p'_{i+1} has Property (c), and for any other vertex taking $u = p'_{i+1}$ (which must be contained in any F') and $F'' = F'^-_{i+1}$ shows that it does not have Property (c). Finally, it remains to note that every hyperedge which contains $p'_1 = p_1$ and p'_0 also contains the whole set L and, in particular, p_0 (similarly, R contains p_{2d}). Hence, also $p_0, p'_1, \ldots, p'_{2d-1}, p_{2d}$ satisfy the assumptions of Rule 7,

and the algorithm finds an application.

As to the running time, there are $|\mathcal{F}^*| \leq m$ possible choices of F, any of the Properties (a)–(c) can be tested for all vertices in $O(m^2 \cdot d^2)$ time by first selecting F' and F'' and then v and u inside them. As successfully testing a condition implies finding one more vertex of $p'_0, \ldots, p'_{2d}$, we test the conditions at most $2d+1$ times in each branch. The sets L and R can be found in $O(m \cdot d)$ time. Finally, the assumptions of Rule 7 for a given sequence $p_0, \ldots, p_{2d}$ can be tested in $O(m \cdot d^2)$ time. Hence the total running time is $O(m \cdot ((2d+1) \cdot m^2 \cdot d^2 + m \cdot d^2)) = O(m^3 d^3)$. □

The description of Rule 7 is quite technical and therefore it seems difficult to obtain an efficient implementation of the rule. This impression is reinforced by the large, although polynomial, running time bound for applying the rule. Nevertheless, it should be possible to apply Rule 7 efficiently. First, the algorithm given above and its analysis is rather simplistic. More sophisticated approaches should yield improvements to the theoretical bounds. Since algorithm engineering is not the main focus of this article and for the sake of clarity, we opted to describe a simple version of the algorithm instead. Second, using a few tricks in searching for hyperedges and vertices that satisfy the desired properties should yield a considerable speed-up in practice. For example, to test for Property (b), it suffices to consider hyperedges F' that are incident with the vertices $p_{i-d}, \ldots, p_i$. Also further properties of the hypergraphs at hand could be taken into account. For example, in some instances of the overlay network application, most of the subscribers consume less than five feeds [24], meaning that the vertices corresponding to the subscribers are contained in at most five hyperedges. For these vertices, testing for Property (b) is then possible in $O(d^2)$ time.

We now derive an upper bound on the number of vertices in instances that are reduced with respect to Rule 4 and Rule 7. As Rule 4 reduces all vertices that have degree one in any solution, the upper bound on the overall number of vertices will be obtained by bounding the number of vertices with degree at least two in the solution. As mentioned above, we will use Rule 7 in Section 5.2, where we also need to bound the number of vertices with degree at least two. Hence we use the following notation.

Definition 5.3 ([27]). *The* 2-core *of a graph G is the uniquely defined induced subgraph of G that has the maximum number of vertices such that each vertex has degree at least two.*

The application of Rule 7 alone does not necessarily yield a bounded-size 2-core in a solution. Instead, the application of Rule 4 for $d \leq 4$ (and of a different rule for $d \geq 5$) "prepares the hypergraph" such that application of Rule 7 yields a size bound on the 2-core of an optimal solution G. We define the notion of being prepared as follows.

Definition 5.4. *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 7, it holds that H and P also satisfy Condition 4.*

The intuition behind the definition is as follows. The first part of the definition guarantees that the solution for a cleared hypergraph consists of the 2-core plus possibly some degree-one vertices attached to this 2-core. The second part of the definition guarantees that any hypergraph with long paths in the 2-core of its solution can be reduced further by applying Rule 7.

For $d \leq 4$ it is sufficient to apply Rule 4 in order to clear a hypergraph.

Lemma 5.5. *Let $H = (V, \mathcal{F})$ be a hypergraph with $d \leq 4$ that is reduced with*

respect to Rule 4. Then, H is cleared.

Proof. The reducedness of H directly implies that there is a solution without any degree-one vertices, hence, the first property of being cleared is satisfied. For the second property, consider P and $\mathcal{F}$ as in Rule 7 and assume that there is a hyperedge $F \in \mathcal{F}$ such that $F \cap P = \{p_i\}$ for some $1 < i < 2d$. Then, by Condition 3 we have $\mathcal{F}(p_i) \supseteq \mathcal{F}(u)$ for each $u \in F \setminus P$ and, hence, Rule 4 applies, a contradiction. □

We now bound the size of reduced instances. We also use this bound in Section 5.2 and, hence, prove it in a slightly more general form than needed for $d \leq 4$.

LEMMA 5.6. *Let (H, k) be a yes-instance of* SID *such that H is connected, cleared, and reduced with respect to Rule 7. Then, there is a solution $G = (V, E)$ for (H, k) such that the 2-core of G has at most $\max\{(9d-1)(\phi-1), (3d-1)\phi\}$ vertices and, hence, at most $9d \cdot \phi$ edges.*

Proof. Among all optimal solutions G for (H, k) such that each vertex of degree at least two is in the 2-core of G, choose G such that it contains the maximum number of degree-one vertices. Now consider the 2-core G' of G. Note that G and G' have the same feedback edge number ϕ. We show a bound on the number of vertices in G' which then gives a bound on the number of edges in G'.

Let $V_{\geq 3}$ denote the vertices with degree at least three in G' and V_2 the degree-two vertices in G'. We first bound the number of components in $G'[V_2]$. If $V_{\geq 3} = \emptyset$, then, clearly, we have at most one component. Otherwise, consider the graph G^* with loops and parallel edges obtained from G' by replacing each maximal path with inner vertices in V_2 by a single edge. The number of edges in G^* is an upper bound on the number of components in $G'[V_2]$. The number of edges in G^* is

$$|V_{\geq 3}| - 1 + \phi = \sum_{v \in V_{\geq 3}} \deg_G(v)/2 \geq 3|V_{\geq 3}|/2.$$

The above relation implies that $|V_{\geq 3}| \leq 2\phi - 2$. Thus, the number of edges in G^* and the number of components in $G'[V_2]$ is at most $\max\{1, 3\phi - 3\}$.

We now show that each connected component of $G'[V_2]$ contains fewer than $3d$ vertices. Consider a connected component C of $G'[V_2]$ with $c+1 \geq 3d$ vertices. Since C is a path, its vertices admit an ordering $p_0, p_1, \ldots, p_c$ with $\{p_i, p_j\} \in E \iff j = i+1$ for all $0 \leq i \leq j \leq c$. Let $P := \{p_0, p_1, \ldots, p_{2d}\}$ and let $\mathcal{F}' := \bigcup_{p \in P} \mathcal{F}(p)$. We show that Rule 7 applies to H, contradicting its reducedness.

First, assume that Condition 2 of Rule 7 does not hold for some $F \in \mathcal{F}'$. That is, there are $0 \leq i < j < \ell \leq 2d$ such that $p_i, p_\ell \in F$ and $p_j \notin F$. Since $c + 1 \geq |P| + d$, a shortest p_i-p_ℓ-path in $G - \{p_j\}$ contains at least $d + 1 > |F|$ vertices. Thus, $G[F]$ is not connected, contradicting G being a solution for (H, k). Hence, Condition 2 of Rule 7 is satisfied.

Now, Condition 1 of Rule 7 can be seen as follows. Assume that there is a pair p_i, p_j, $i < j$, of vertices in C such that p_j covers p_i. By the above, all hyperedges that contain p_i and p_j also contain $\{p_{i+1}, \ldots, p_{j-1}\}$. Hence, p_{i+1} also covers p_i. Since p_i and p_{i+1} are adjacent in G, this implies that a new solution can be obtained by making all neighbors (except p_{i+1}) of p_i adjacent to p_{i+1} instead. The new graph is also a solution, and has one additional degree-one vertex. This contradicts our choice of G. Hence, Condition 1 of Rule 7 holds.

Let $F \in \mathcal{F}$ with $F \cap \{p_0, p_{2d}\} = \emptyset$ and note that $G[F]$ is connected. Since H is cleared, and since there is some $p_j \in F \cap P$, we know that $G[F]$ consists entirely of

vertices in P plus some degree-one vertices. Each degree-one vertex v is covered by its neighbor $p \in P$ in G. Thus Condition 3 of Rule 7 holds.

Finally, Condition 4 of Rule 7 follows since H is cleared. Thus, Rule 7 applies, a contradiction.

By the above, each connected component in $G' - V_{\geq 3}$ has less than $3d$ vertices. Hence, $|V_2| \leq (3d-1)\max\{3\phi - 3, 1\}$. Altogether, this implies

$$|V_2| + |V_{\geq 3}| \leq (3d-1)\max\{3\phi-3,1\} + 2\phi - 2 = \max\{(9d-1)(\phi-1), (3d-1)\phi\}.$$

By definition of ϕ this means that the 2-core of G has at most $9d \cdot \phi$ edges. ∎

For $d \leq 4$, after applying Rule 4 the size bound on the 2-core immediately gives a bound on the overall instance size.

THEOREM 5.7. *An instance of* SUBSET INTERCONNECTION DESIGN *with maximum hyperedge size $d \leq 4$ can be reduced to an equivalent one with at most* $\max\{35(\phi-1), 11\phi\}$ *vertices in $O(n \cdot m^3)$ time.*

Proof. The kernelization algorithm exhaustively applies Rule 4 and Rule 7. Reducedness with respect to Rule 4 ensures that there are no degree-one vertices in any optimal solution and, by Lemma 5.5, that the hypergraph is cleared. Consequently, being reduced with respect to Rule 7 implies that any solution contains at most $\max\{(9d-1)(\phi-1), (3d-1)\phi\}$ vertices of degree at least two. Altogether this implies the bound on the number of vertices.

For the running time, we apply Rule 4 exhaustively, then apply Rule 7 exhaustively, and repeat until neither applies anymore. Since each application of one of the rules deletes at least one vertex, we iterate at most n times. Applying the running time bounds given by Lemmata 3.4 and 5.2 we obtain the overall running time bound. ∎

5.2. A Fixed-Parameter Algorithm for ϕ and d. Our polynomial-time data reduction in the last section does not generalize to arbitrary d. The reason is that the condition $|F| \leq 4$ is necessary for the correctness of Rule 4 or, equivalently, that Rule 1 is incorrect if $d > 5$. However, using an additional reduction rule (Rule 8 below), we obtain the same vertex-bound on 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 fact that the 2-core of a solution has bounded size we obtain a branching algorithm with running time $O(d^{18d\phi} \cdot d \cdot n \cdot m + n \cdot m^3 \cdot d^2)$. The algorithm first applies Rule 7 and Rule 8 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 8 to clear the input hypergraph and to make Lemma 5.6 applicable.

RULE 8. *Let $H = (V, \mathcal{F})$ be a hypergraph, $F \in \mathcal{F}$, and $F = \{u, u_1, \ldots, u_\ell\}$ such that u covers each u_i. Then, remove the vertices $u_1, \ldots, u_\ell$ from H and decrease k by ℓ.*

LEMMA 5.8. *Rule 8 is correct and one application takes $O(n \cdot m \cdot d)$ time.*

Proof. We first show the correctness of the rule. We show that there is an optimal solution G such that $G[F]$ is a star with center vertex u. Let G' be any optimal solution for H and assume that there are two adjacent vertices in F none of which is u. Note that we may choose two such adjacent vertices u_i, u_j such that $\{u, u_j\} \subseteq N_{G'}(u_i)$; this is possible because otherwise $G'[F]$ is not connected. Remove the edge $\{u_i, u_j\}$ from G' and add $\{u, u_j\}$ to obtain G. We prove that the graph G is a solution.

Consider any hyperedge $F' \in \mathcal{F}$ and any path P' between two vertices in $G'[F']$. If P' contains the edge $\{u_i, u_j\}$, then we may replace it by $\{u_i, u\}, \{u, u_j\}$ to obtain the walk P in G. Since u covers both u_i and u_j the path P is contained in $G[F']$. Thus, $G[F']$ is connected for each $F' \in \mathcal{F}$ meaning that G is an optimal solution. By repeating the replacement of edges described above, we may arrange that $G[F]$ is a star.

Now, note that Rule 8 is basically a series of applications of Rule 2. Clearly, Rule 2 applies to u_1, so one can safely remove u_1 and set $k := k - 1$. Afterwards, Rule 2 still applies to u_2, so one can remove u_2 and decrease k once more. This can be repeated until all u_i's are removed from H.

The running time of the rule can be seen as follows. First, we construct the covering graph G_C for H in $O(n \cdot m \cdot d)$ time using Lemma 2.1. Then, for each hyperedge $F \in \mathcal{F}$ we check whether there is a vertex u such that there are arcs (u, v) in G_C for every $v \in F \setminus \{u\}$. If so, then we remove each vertex in $F \setminus \{u\}$. This costs $O(m \cdot d^2)$ time. It is easy to check that this procedure finds an application of Rule 8 if there is one. ∎

For $d \geq 5$, we can now use Rule 8 to clear hypergraphs.

LEMMA 5.9. *Let $H = (V, \mathcal{F})$ be a hypergraph that is reduced with respect to Rule 8. Then, H is cleared.*

Proof. We first prove the first statement of being cleared, namely, that there is an optimal solution G for H such that each vertex not in the 2-core of G has degree one. We use the notion of "pending trees". Let G be a graph. If $G[V']$ is the 2-core of G, then a *pending tree* of G is a connected component of $G[V \setminus V']$ plus its unique neighbor in V'.

Pick an optimal solution G and consider its pending trees and their vertex sets $C_1, \ldots, C_c$. For each pending tree C_i there is a unique vertex x both in C_i and in the 2-core of G. Denote $x =: \text{root}(C_i)$. If the 2-core of G is empty, then there is only one pending tree C_1 and we choose $\text{root}(C_1)$ to be an arbitrary vertex of degree one instead. Next, consider optimal solutions that contain the maximum number of degree-one vertices. Among these optimal solutions, choose G such that

$$\text{val}(G) := \sum_{i=1}^{c} \sum_{\substack{v \in C_i \\ \deg_G(v)=1}} \text{dist}(\text{root}(C_i), v)$$

is minimum, where $\text{dist}(u, v)$ is the length of a shortest path between u and v in G.

Assume now that there is a pending tree with vertex set $C \subseteq V$ such that $G[C]$ contains at least one vertex with degree at least two in $G[C]$. Choose $u \in C$ with degree at least two such that a shortest path between u and $\text{root}(C)$ has maximum length. Consider the neighbors of u in G. Let v be the neighbor of u on the shortest path from u to $\text{root}(C)$. All the neighbors of u different from v must be of degree one due to the choice of u according to the maximum length path to $\text{root}(C)$. Let us call them $u_1, \ldots, u_\ell$. By Observation 1, $\mathcal{F}(u) \supseteq \mathcal{F}(u_i)$ for all $i \in \{1, \ldots, \ell\}$. Consider some u_i. Since H is reduced with respect to Rule 8, there is no subset U of the degree-one neighbors of u and no hyperedge $F \in \mathcal{F}$ such that $F = U \cup \{u, u_i\}$. Hence, each hyperedge incident with u_i also contains some vertex other than u and its degree-one neighbors. We conclude that each hyperedge $F \in \mathcal{F}$ that contains u_i also contains v since, otherwise $G[F]$ is not connected. Thus, the graph G' obtained from G by removing the edge $\{u_i, u\}$ and adding the edge $\{u_i, v\}$ is an optimal solution. However, the distance of u_i to $\text{root}(C)$ is smaller in G' than in G. Hence, either u_i is

the only degree-one neighbor of u in G, contradicting the choice of G according to the maximum number of degree-one vertices, or G' exhibits a smaller $\mathrm{val}(G')$ contradicting the choice of G according to the minimum $\mathrm{val}(G)$. Hence, there are no pending trees C that contain degree-two vertices and the first statement of the cleared-definition now follows.

It remains to prove the implications of the conditions from Rule 7. Let H, $P := \{p_0, \ldots, p_{2d}\}$ with $P \subseteq V$ and $\mathcal{F}' := \bigcup_{p \in P} \mathcal{F}(p)$ satisfy the Conditions 1 to 3 of Rule 7. If there is a hyperedge $F \in \mathcal{F}$ such that $F \cap P = \{p_i\}$ for some $0 < i < 2d$, then we have $\mathcal{F}(u) \subseteq \mathcal{F}(p_i)$ for every $u \in F \setminus \{p_i\}$ by Condition 3, and Rule 8 applies to $p_i, u_1, \ldots, u_\ell$, where $\{p_i, u_1, \ldots, u_\ell\} = F$. Hence, if H is reduced with respect to Rule 8, then Condition 4 is satisfied. ∎

Now, Lemma 5.6 is applicable to hypergraphs that are reduced with respect to Rule 8, that is, we can reduce any input instance in polynomial time to one such that there is a solution with at most $9d \cdot \phi$ edges in the 2-core. Based on this fact, we devise a branching algorithm for the parameter (d, ϕ). 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 ϕ, we ensure that the search tree has depth at most $9d \cdot \phi$ 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 8.

Branching Rule 1. *Let F be a hyperedge of H such that, with $F_0 \subseteq F$ denoting the vertices of F whose degree in G is zero, $G[F]$ is disconnected and cannot be made connected by, for each $u \in F_0$, adding 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 $G[F]$, decrementing k' by one.*

Lemma 5.10. *Branching Rule 1 is correct, that is, the original instance is a yes-instance if and only if at least one of the created instances is a yes-instance.*

Proof. "⇐": Any solution for a yes-instance created by the branching is also a solution for the original instance.

"⇒": We show that there is an optimal solution G' of the original instance such that the 2-core of G' contains an edge that is in $G'[F]$ but not in $G[F]$. Recall that we assume H to be reduced with respect to Rule 8. Hence, H is cleared and we can assume that all edges of $G'[F]$ that are not in the 2-core of G' are incident with degree-one vertices in G'. Assume towards a contradiction that all edges in $G'[F]$ that are not in $G[F]$ are not in the 2-core of G'. Thus, they are incident with degree-one vertices in G'. Hence, all vertices in F_0 have degree one in G'. Since $G'[F]$ is connected, each vertex $u \in F_0$ is connected to some $v \in F \setminus F_0$. By Observation 1, v covers u and, thus, the condition of Branching Rule 1 is not satisfied. ∎

Next, we show that, if Branching Rule 1 does not apply, meaning that its preconditions are not fulfilled for any choice of hyperedge F, then we can solve the instance by greedily assigning the remaining vertices.

Lemma 5.11. *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 $O(n \cdot d \cdot m)$ time.*

Proof. Let $H = (V, \mathcal{F})$, let V_0 be the set of degree-zero vertices in G, and let G' be obtained from G by adding exactly one edge $\{u, v\}$ to G for each $u \in V_0$ where $v \in V \setminus V_0$ is an arbitrary vertex covering u. Note that G' can be computed in $O(n \cdot d \cdot m)$ time,

by first computing the covering graph of H using Lemma 2.1 and then iterating over each vertex and adding the appropriate edges. We now show that G' is an optimal solution.

We first show that for each hyperedge $F \in \mathcal{F}$, the graph $G'[F]$ is connected. Let $F_0 = F \cap V_0$ for a hyperedge $F \in \mathcal{F}$. Since Branching Rule 1 does not apply to the instance, $G[F]$ can be made connected by adding an edge for each $u \in F_0$ between u and some vertex $w \in F \setminus F_0$ that covers u. Thus, because each $u \in F_0$ gets at most one incident edge in this way, $G[F \setminus F_0]$ is connected. Since, for each $u \in F_0$, its neighbor v in G' covers u and we know that $v \in F \setminus F_0$, also $G'[F]$ is connected.

It remains to show that G' is optimal. Note that G' has exactly $|V_0|$ more edges than G. Let G^* be any solution that is a supergraph of G. Now merge in G^* all vertices in $V \setminus V_0$ into one vertex. The resulting graph contains at least $|V_0|$ edges, since G^* is connected. Hence, G^* has $|V_0|$ edges which are incident with vertices from V_0. Thus, at least $|V_0|$ edges have to be added to G' to obtain a solution. ∎

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

THEOREM 5.12. SUBSET INTERCONNECTION DESIGN *can be solved in* $O(d^{18d\phi} \cdot d \cdot n \cdot m + n \cdot m^3 \cdot d^2)$ *time.*

Proof. Let $H = (V, \mathcal{F})$ be the input hypergraph. The branching algorithm works as follows. It starts by exhaustively applying Rule 8, then exhaustively applying Rule 7 and repeating until neither applies anymore. Since each application removes at least one vertex the applications of Rule 8 take $O(n \cdot (d \cdot n \cdot m))$ time by Lemma 5.8. The applications of Rule 7 take $O(n/d \cdot (m^3 \cdot d^3)) = O(n \cdot m^3 \cdot d^2)$ time by Lemma 5.2 and the fact that each application removes at least $2d - 2 > d$ vertices. Note that the running time contribution of Rule 7 dominates the one of Rule 8. Now, Lemma 5.9 and Lemma 5.6 imply that there is a solution whose 2-core has at most $9d \cdot \phi$ edges for a yes-instance. Hence, it is correct to initially search for a solution of H that is a supergraph of the edgeless graph on V and has at most $k' := 9d \cdot \phi$ edges in the 2-core. Then, we apply Branching Rule 1 as often as possible. Each application creates at most $\binom{d}{2}$ branches, and in each recursive branch the value of k' is decreased by one. The correctness of this branching follows by Lemma 5.10. Note that if $k' < 0$, then we can safely abort the search at the current search tree node, since there is no solution satisfying its constraints. Finally, if Branching Rule 1 does not apply, we can compute an optimal solution satisfying the constraints of the search tree node in $O(n \cdot d \cdot m)$ time (by Lemma 5.11).

To check whether Branching Rule 1 applies, we maintain the covering graph throughout the search tree. At each node, we iterate over each edge F, add the edges between vertices in F and arbitrary covering vertices to G, and check whether the result is connected. This needs $O(m \cdot (d + d^2))$ time at each node. Initializing the covering graph can be done in $O(d \cdot n \cdot m)$ time (Lemma 2.1). The size of the search tree is at most $\binom{d}{2}^{9d\phi} \leq d^{18d\phi}$ and hence the overall running time is $O(n \cdot m^3 \cdot d^2 + d \cdot n \cdot m + d^{18d\phi} \cdot (m \cdot d^2 + n \cdot m)) = O(d^{18d\phi} \cdot d \cdot n \cdot m + n \cdot m^3 \cdot d^2)$. ∎

6. Conclusion. We contributed to a better understanding of a natural NP-hard network construction problem which has been previously studied in many application contexts (also independently from each other and under different names). In doing so, we revealed an incorrectness in previous work, refined the complexity analysis of the problem, and, in particular, provided a thorough investigation concerning its amenability to efficient preprocessing.

The linear-time algorithm for a constant number m of hyperedges that follows

from our data reduction rules seems foremost of theoretical interest. In practice, it seems unlikely that the data reduction rules apply often because of the large twin classes they require which are of size exponential in m. An interesting task for future research is thus to improve the bounds on the twin classes. However, we conjecture that an upper bound polynomial in m on the size of the whole remaining instance, that is, a polynomial-size problem kernel, cannot be achieved.

Of more practical value seem to be Rule 7 as well as Rules 2 to 4, 6, and 8 which repair the incorrect Rule 1 from the literature [14, 18]. Indeed, a case study of when Rule 1 applies, but none of the repaired rules do, would be useful and could lead to further relevant data reduction rules.

We showed that finding tree-like solutions in hypergraphs with small hyperedges is tractable. Here, algorithm engineering is needed to improve the running times and to merge our reduction rules with existing solution strategies. On the theoretical side, we did not resolve yet whether Subset Interconnection Design is fixed-parameter tractable with respect to the feedback edge number ϕ of the solution *alone*. A possible line of attack would be to first find out whether, in our results, we can replace the parameter d (maximum hyperedge size) by the smaller parameter "maximum overlap number", that is, the maximum size of an intersection between any two input hyperedges. 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.

More generally, it seems also promising to consider data reduction for variants of Subset Interconnection Design that ask to minimize the maximum degree instead of the average degree (see Onus and Richa [26]), and the various variants occurring in hypergraph drawing [2, 3, 19, 20]. It is furthermore of practical interest to deal with edge weights for the constructed network [22]; 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 data-driven parameter analysis for real-world instances, is promising from a practical and from a theoretical side.

Acknowledgments. We thank Peter Damaschke (Chalmers University, Göteborg) for stimulating discussions and for pointing us to the Subset Interconnection Design problem. We are grateful to Falk Hüffner for pointing us to the Minimum Topic-Connected Overlay problem. We also thank the anonymous reviewers of SIDMA for their helpful comments in improving our paper.

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] U. Brandes, S. Cornelsen, B. Pampel, and A. Sallaberry. Blocks of hypergraphs - Applied to hypergraphs and outerplanarity. In *Proc. 21st IWOCA*, volume 6460 of *LNCS*, pages 201–211. Springer, 2011.

[3] U. Brandes, S. Cornelsen, B. Pampel, and A. Sallaberry. Path-based supports for hypergraphs. *J. Discrete Algorithms*, 14:248–261, 2012.

[4] K. Buchin, M. J. van Kreveld, H. Meijer, B. Speckmann, and K. Verbeek. On planar supports for hypergraphs. *J. Graph Algorithms Appl.*, 15(4):533–549, 2011.

[5] C. Bujtás, Z. Tuza, and V. Voloshin. Color-bounded hypergraphs, V: Host graphs and subdivisions. *Discuss. Math. Graph Theory*, 31(2):223–238, 2011.
[6] 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.
[7] V. Conitzer, J. Derryberry, and T. Sandholm. Combinatorial auctions with structured item graphs. In *Proc. 19th AAAI*, pages 212–218. AAAI Press / The MIT Press, 2004.
[8] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.
[9] D.-Z. Du. An optimization problem on graphs. *Discrete Appl. Math.*, 14(1): 101–104, 1986.
[10] D.-Z. Du and D. F. Kelley. On complexity of subset interconnection designs. *J. Global Optim.*, 6(2):193–205, 1995.
[11] D.-Z. Du and Z. Miller. Matroids and subset interconnection design. *SIAM J. Discrete Math.*, 1(4):416–424, 1988.
[12] D.-Z. Du, B. Gao, and W. Wu. A special case for subset interconnection designs. *Discrete Appl. Math.*, 78(1-3):51–60, 1997.
[13] H. Fan and Y.-L. Wu. Interconnection graph problem. In *Proc. FCS 2008*, pages 51–55. CSREA Press, 2008.
[14] 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.
[15] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.
[16] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007.
[17] M. Habib, C. Paul, and L. Viennot. Partition refinement techniques: An interesting algorithmic tool kit. *Int. J. Found. Comput. Sci.*, 10(2):147–170, 1999.
[18] 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.
[19] D. S. Johnson and H. O. Pollak. Hypergraph planarity and the complexity of drawing Venn diagrams. *J. Graph Theory*, 11(3):309–325, 1987.
[20] M. Kaufmann, M. J. van Kreveld, and B. Speckmann. Subdivision drawings of hypergraphs. In *Proc. 16th GD*, volume 5417 of *LNCS*, pages 396–407. Springer, 2008.
[21] B. Klemz, T. Mchedlidze, and M. Nöllenburg. Minimum tree supports for hypergraphs and low-concurrency euler diagrams. In *Proc. 14th SWAT*, volume 8503 of *LNCS*, pages 265–276. Springer, 2014.
[22] E. Korach and M. Stern. The clustering matroid and the optimal clustering tree. *Math. Program.*, 98(1-3):385–414, 2003.
[23] D. Král, J. Kratochvíl, and H.-J. Voss. Mixed hypercacti. *Discrete Math.*, 286 (1-2):99–113, 2004.
[24] H. Liu, V. Ramasubramanian, and E. G. Sirer. Client behavior and feed characteristics of RSS, a publish-subscribe system for web micronews. In *Proc. 5th IMC*, pages 29–34. USENIX Association, 2005.
[25] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.
[26] M. Onus and A. W. Richa. Minimum maximum-degree publish-subscribe overlay

network design. *IEEE/ACM Trans. Netw.*, 19(5):1331–1343, 2011.

[27] S. B. Seidman. Network structure and minimum degree. *Soc. Networks*, 5:269–287, 1983.

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

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

Appendix A. A Family of Instances Requiring a Solution with at least $2^{\Omega(m)} + n$ Edges.

In this section we show that the bound given by Theorem 4.1 and used in Section 4 is optimal in the sense that it cannot be improved to a bound polynomial in m.

To this end, we construct a hypergraph H with $m = 2q+1$ hyperedges $F_1, \ldots, F_{2q+1}$, where $q \geq 2$, such that each vertex is in exactly q hyperedges and each intersection of any q hyperedges consists of exactly one vertex. Therefore we have $n = \binom{2q+1}{q}$ vertices $v_1, \ldots, v_n$, and each of them can be described by a vector in $\{0,1\}^{2q+1}$ with exactly q one-entries. Hence, a hyperedge F_i consists of those vertices whose vector description has a one-entry in the ith position and $q-1$ one-entries in the other m positions. This sums up to $\binom{2q}{q-1}$. Hence, each hyperedge has exactly $\binom{2q}{q-1}$ vertices. This completes the construction. Note that any solution for hypergraph H must contain at least $\binom{2q}{q-1} - 1$ edges if restricted to some hyperedge.

Suppose that graph G is a solution for H. If each edge in G is contained in exactly one induced subgraph $G[F_i]$ for some hyperedge F_i, then G would need $(2q+1)\cdot(\binom{2q}{q-1}-1)$ edges. In fact, each edge $\{v_x, v_y\}$ in G can be contained in at most $q-1$ subgraphs $G[F_i]$, $1 \leq i \leq m$ (these correspond to the positions where both v_x and v_y have one-entries in their vector representations). Therefore, each solution contains at least $k_{\text{opt}} \geq \frac{2q+1}{q-1} \cdot \left(\binom{2q}{q-1} - 1\right)$ edges. Now we compare this to the number of vertices. We have

$$
\begin{aligned}
k_{\text{opt}} - n &\geq \frac{2q+1}{q-1} \cdot \left(\binom{2q}{q-1} - 1\right) - \binom{2q+1}{q} \\
&= \frac{1}{q-1}\left(\frac{q\cdot(2q+1)!}{q!\cdot(q+1)!} - \frac{(q-1)\cdot(2q+1)!}{q!\cdot(q+1)!}\right) - \frac{2q+1}{q-1} \\
&= \frac{1}{q-1}\left(\frac{(2q+1)!}{q!\cdot(q+1)!}\right) - \frac{2q+1}{q-1} \\
&= \frac{1}{q-1}\binom{2q+1}{q} - \frac{2q+1}{q-1} \\
&= \frac{2q+1}{(q-1)(q+1)}\binom{2q}{q} - \frac{2q+1}{q-1} \\
&\geq \frac{2q+1}{(q-1)(q+1)} \cdot \frac{2^{2q-1}}{\sqrt{q}} - 5 \\
&> \frac{2^{2q}}{(q+1)\sqrt{q}} - 5.
\end{aligned}
$$

The last but one inequality holds due to the Stirling's approximation and $q \geq 2$. Thus, $k_{\text{opt}} \geq 2^{\Omega(q)} + n = 2^{\Omega(m)} + n$.