On the Effectiveness of Data Reduction for Covering Problems in Real-World Transit Networks

Über die Effektivität von Datenreduktion für Überdeckungsprobleme in Echtwelt-Vekehrs-Netzwerken

Master's Thesis

submitted in partial fulfillment for the degree of

Master of Science

in IT-Systems Engineering

Submitted by: **Philipp Fischbeck**

Supervisor: Prof. Dr. Tobias Friedrich
Advisors: Dr. Thomas Bläsius
Martin Schirneck

of the Chair for Algorithm Engineering

Potsdam, January 31 2018

Abstract

Given a graph and a set of paths, we want to find the minimal set of vertices such that each path is covered by at least one chosen vertex. Although this problem is NP-hard, real-world instances can be solved almost completely by a set of simple reduction rules. We examine this behavior from a theoretical and empirical perspective. First, we show that the problem is easy to solve for forests and cycle graphs. However, the problem is NP-hard for a feedback vertex number of 2 and a treewidth of 3. This indicates that the explanation for the effectiveness does not lie in the graph representation of problem instances. Thus, we examine the Hitting Set problem that arises when ignoring the graph representation and interpreting a path as a mere set of vertices. Through this relation, we show that the problem remains NP-hard even for very strong restrictions. Hitting Set instances that have a representation as a path graph can be recognized as such in polynomial time. However, finding the graph representation with the fewest edges is NP-hard.

Based on the analysis of publicly available transit datasets, we show that the real-world instances are clustered and have heterogeneous stations, with the number of lines per station distributed according to a power law. We describe a model to generate random problem instances with adjustable clustering and heterogeneity. We use this model to show that while the heterogeneity does positively influence the effectiveness of the reduction rules, the largest effect comes from the clustering.

Lastly, we show a strong relation between the reduction rules for the Hitting Set problem and reduction rules for the Maximum Independent Set problem on the intersection graph of the family of sets. We prove that the size of any independent set is a lower bound on the size of the maximum hitting set and show that the two bounds are very close for real-world instances. We show that the reduction rules need to be effective for Maximum Independent Set in order for them to be effective for Hitting Set.

Zusammenfassung

In einem gegebenen Graphen mit einer Menge von ausgezeichneten Pfaden möchten wir eine minimale Menge von Knoten finden, sodass jeder Pfad von mindestens einem gewählten Knoten abgedeckt wird. Obwohl dieses Problem NP-schwer ist, können Echtwelt-Instanzen mithilfe von einfachen Reduktionsregeln fast vollständig gelöst werden. Wir untersuchen dieses Verhalten aus einer theoretischen und empirischen Perspektive. Zunächst zeigen wir, dass das Problem für Wälder und Kreise effizient lösbar ist. Andererseits bleibt das Problem NP-schwer, selbst wenn man die Eingabe beschränkt auf Graphen mit einer Größe des Feedback Vertex Set von 2 oder einer Baumweite von 3. Dies deutet darauf hin, dass die Erklärung für die Effektivität nicht in der Graphen-Repräsentation von Probleminstanzen zu finden ist. Daher untersuchen wir das Hitting Set-Problem, welches entsteht, wenn wir die Graphen-Repräsentation ignorieren und einen Pfad als Menge von Knoten interpretieren. Durch diese Relation zeigen wir, dass das Problem selbst bei starken Einschränkungen NP-schwer bleibt. Man kann in polynomieller Zeit erkennen, ob eine Hitting Set-Instanz eine Repräsentation als Pfad-Graph hat. Jedoch ist das Finden einer Graphen-Repräsentation mit möglichst wenigen Kanten NP-schwer.

Basierend auf einer Analyse von frei verfügbaren Verkehrs-Netzwerken zeigen wir, dass Echtwelt-Instanzen geclustert sind und heterogene Haltestellen haben, wobei die Anzahl von Linien pro Haltestelle nach einem Potenzgesetz verteilt ist. Wir beschreiben ein Modell zur Generierung zufälliger Probleminstanzen mit variablem Clustering und Heterogenität. Wir benutzen dieses Modell, um zu zeigen, dass die Heterogenität zwar einen positiven Einfluss auf die Effektivität der Reduktionsregeln hat, der stärkste Einfluss jedoch vom Clustering kommt.

Schließlich zeigen wir einen starken Zusammenhang zwischen den Reduktionsregeln für das Hitting Set-Problem und Reduktionsregeln des Maximum Independent Set-Problems auf dem Schnittgraph der Familie von Mengen. Wir beweisen, dass die Größe jeder unabhängigen Menge eine untere Schranke an die Größe des größten Hitting Set ist und dass die beiden Schranken für Echtwelt-Instanzen sehr nah sind. Wir zeigen, dass die Reduktionsregeln für Maximum Independent Set effektiv sein müssen, damit sie auch für Hitting Set effektiv sind.

Contents

1 Introduction

Consider the following problem in public transport. We are given a set of stations as well as a set of transit lines going through some of the stations. We want to select stations such that every line goes through at least one of the selected stations. If we aim for a minimum number of selected stations, we can find the stations that are most important for the transit network or choose these stations for strategical placement of, e.g., maintenance facilities of trains. For this problem, we can formulate two simple reduction rules. A station can be removed if there is another station that would cover a superset of the lines of the first station, because the second station is always at least as good of a choice as the first station. Similarly, a line can be removed if there is another line that stops at a subset of the stations that the first line stops at. If our selected stations cover the second line, they are guaranteed to cover the first line too.

We can model the stations and lines as vertices and paths in a graph, and we want to find the minimum set of vertices such that all paths are covered by at least one vertex. This problem is called Path Cover by Vertices. In 1998, Weihe introduced this problem and the above-mentioned reduction rules [38]. In corporation with a subsidiary of the Deutsche Bahn AG, Germany's largest railway company, the author worked on solving this problem based on the timetables of trains in the German railroad network. Surprisingly, the reduction rules turned out to be very effective, completely solving the problem or leaving only a very small problem core that can be solved by brute force. This is especially surprising considering that Path Cover by Vertices is NP-hard by reduction a from Hitting Set, as shown by Weihe [38]. Thus, there is a substantial gap between our theoretical understanding of this problem and the practical results that can be achieved by the reduction rules. Closing this gap can lead to advantages in both ways. First, it would improve our theoretical understanding of the problem, which would also improve our

understanding of related problems such as Hitting Set and Set Cover. Second, the findings could be translated into algorithms that utilize the properties that lead to the effectiveness of the reduction rules, thereby allowing us to tackle the problem in practice in new ways.

The main reason for this gap is that the theoretical analysis considers the worst case, which does not match typical real-world instances. There are two common approaches to closing such a gap. The first approach is the parameterized complexity. The field of parameterized complexity was first systematically researched by Downey and Fellows in 1999 [15]. A problem is called *fixed-parameter tractable* in some parameter k if there is an algorithm that solves the problem in run time $f(k) \cdot n^{O(1)}$, where f is some computable function depending only on k, and n is the problem input size. For a constant k, such an *FPT algorithm* allows for a polynomial run time that is independent of k (as opposed to a run time like $O(n^k)$). One common parameter is the solution size. Other possible parameters describe structural properties of the input data. In the case of Path Cover by Vertices, one such property of the input graph is the treewidth, which models the tree-likeness of a graph. Other possible parameters are the maximum path length or the maximum vertex degree of the input graph. An FPT algorithm in any of these parameters would explain which properties allow for a Path Cover by Vertices problem to be solved efficiently. If real-world instances also have these properties, the gap is closed.

The second approach is to consider an average case rather than the worst case. However, the definition of an average requires a probability distribution over the input. This can be done by considering "typical" instances with the help of a model that accurately depicts key properties of real-world problem instances. As noted by Karp [26], "the approach seems to have considerable explanatory power". More on the field of average-case complexity can be found in the survey by Bogdanov and Trevisan [6]. This approach requires a fine balance between choosing a realistic model that is possibly harder to analyze and choosing a simple model that lacks explanatory power. Therefore, we aim towards a model that captures the key properties of real-world instances. In the case of our transit networks in particular, candidates for key properties are heterogeneity and clustering. Heterogeneity means there are few stations with many lines going through them (central stations) and many stations with only a handful of lines. Many large networks from various domains, including biological and social networks, are known to be heterogeneous [12, 24], which usually originates in the large differences in importance of the entities represented in the network. This heterogeneity often follows a power-law distribution in the

degrees of the entities [17].

In graphs, clustering means that vertices with a common neighbor are likely to be neighbors with each other. Many real-world networks exhibit a kind of clustering or transitivity [37, 22]. The canonical notion is the *global clustering coefficient*, which is the ratio of the number of triangles to the number of unordered connected triplets of vertices. This notion is not applicable in two-mode networks such as our Hitting Set instances, since there are no cycles of odd length. Lapaty et al. give an overview of the classical notions and their variants as well as modifications that are suited for bipartite graphs [29]. We will discuss some of these adjusted parameters in detail in Section 4.1.

Erdős and Rényi laid the foundation for the generation of random graphs with the Erdős-Rényi model in 1959 [16]. A graph with n vertices is generated by connecting each pair of vertices independently with probability p. While this model is very simple and therefore easier to analyze, it lacks the properties of heterogeneity and clustering. Numerous models have been introduced to account for heterogeneity. One such model is the Barabási-Albert model [5]. In this model, vertices are gradually added, and they are connected to already existing vertices with a probability proportional to their degree, creating a *preferential attachment*. Another approach can be found in the Chung-Lu model [11], in which vertices are assigned weights, and two vertices are connected with a probability proportional to the product of their weights. While both models allow for the generation of heterogeneous graphs, they are not clustered. One approach to generating graphs with high clustering is to use an underlying geometry. Random coordinates in an Euclidean geometry are assigned to each vertex, and two vertices are connected if and only if they are close. Again, while this approach yields high clustering [32], it does not yield a heterogeneous graph. The model of *hyperbolic random graphs* introduced by Krioukov et al. [28] leads to graph with both of these properties. In this model, the Euclidean geometry is replaced by a hyperbolic plane. This approach inherits the clustering from the geometric approach while also displaying heterogeneity. Bringmann et al. introduce the model of geometric inhomogeneous random graphs (GIRGs) [8], which can be seen of a combination of the Chung-Lu model with geometry. It also generates graphs which are both heterogeneous and clustered and is in fact a generalization of the hyperbolic model.

In this thesis, we tackle the problem with both of these approaches; we evaluate possible parameters for a parameterized complexity, and we conduct an empirical average-case analysis based on an instance generation model. We find that the graph structure of a Path Cover by Vertices instance can

not be used as grounds for an explanation of the effectiveness of the reduction rules. A parameterization in many parameters, including the solution size and the treewidth, is impossible unless P=NP. Therefore, in order to understand the Path Cover by Vertices problem, we should understand the underlying Hitting Set problem. Furthermore, we present a model to generate Hitting Set instances that has structural properties similar to real-world instances. In our empirical average-case analysis of this model, we show that the properties of clustering and heterogeneity have a strong correlation to the effectiveness of the reduction rules. Both of these properties are also present in real-world instances of transit networks. We also present an interesting relation between the Hitting Set problem and the Maximum Independent Set problem, showing that a better understanding of the reduction rules of the Maximum Independent Set problem might also lead to a better understanding of the Hitting Set problem.

1.1 Outline

In Chapter 2 we define the notions that will be used throughout this thesis as well as the Path Cover by Vertices problem and its reduction rules. As we show in Chapter 3, the usual parameterized approach based on the underlying graph does not yield a satisfying explanation for the effectiveness of the reduction rules. While Path Cover by Vertices is easy to solve for forests and cycle graphs, a generalization of closeness to trees fails for several metrics common in graph theory, in particular the feedback vertex number and the treewidth. We give an overview of previous research on the underlying Hitting Set problem in Section 3.2. We apply these results to the Path Cover by Vertices problem and find that they also do not yield an explanation for the effectiveness of the reduction rules. In Chapter 4, we introduce a model for generating Hitting Set instances that allow for configurable heterogeneity and clustering. We show that both of these structural properties are present in real-world instances, and both properties strongly correlate to the effectiveness of the rules. Furthermore, we compare several parameters for measuring the clustering of a Hitting Set instance. In Chapter 5, we show an interesting relation between the reduction rules of Hitting Set and those of a related problem, namely Maximum Independent Set. An independent set of a graph is a set of vertices such that none of them share any edges. Similarly, we can look for a maximum set of sets such that none of the sets share any elements. Any such independent set of sets forms a lower

bound on the best hitting set, and in fact, all hitting set reduction rules are valid independent set reduction rules too. Through this connection and the observation that the two solution sizes are very close for real-world instances, we provide a foundation for future work in the understanding of the reduction rules. Finally, we conclude our work in Chapter 6 and give pointers for possible future research.

2 Preliminaries

2.1 Graphs

A *simple graph* G (*graph* for short) is defined on a set of vertices V and a set of edges $E \subseteq \{\{u, v\} \mid u, v \in V \wedge u \neq v\}$. A *subgraph* of a graph is given by a subset of the vertices and edges of a graph, where for any edge $\{u, v\}$ in the subgraph, both u and v have to be in the subgraph as well. Two vertices u, v are *adjacent* if $\{u, v\} \in E$. For a vertex v, the set of *neighbors* $\{u \mid \{u, v\} \in E\}$ is called the *neighborhood* of v. The number of neighbors of a vertex is called its *degree*. A graph is *planar* if there is a way of drawing the graph in the plane such that no two edges cross each other. A *path* is a sequence of vertices such that consecutive vertices always share an edge, with no vertex occurring more than once. The *length* of a path is the number of edges used. A *cycle* is a sequence of adjacent vertices where all vertices are distinct, with the exception that the first and last vertex have to be identical. A graph is *cycle-free* if it has no cycle as a subgraph. A graph is called *connected* if there is a path between any pair of vertices. A *component* of a graph is a maximal connected subgraph, meaning no other vertex can be added without violating the connectedness. A cycle-free connected graph is called a *tree*. A cycle-free graph is called a *forest*. Each of its components forms a tree. In a forest or tree, vertices of degree 1 are called *leaves*.

Two parameters that capture the closeness of a graph to a tree are the feedback vertex number and the treewidth of a graph. A subset of the vertices of a graph is called a *feedback vertex set* if the removal of these vertices yields a cycle-free graph. The *feedback vertex number* of a graph is the size of its minimum feedback vertex set. A *tree decomposition* of a graph $G = (V, E)$ is a tree on so-called bags $X_1, \ldots, X_n \subseteq V$ such that every vertex occurs in some bag, for every edge, some bag contains both vertices of that edge, and for every vertex, the bags that contain that vertex form a subtree in the tree decomposition.

The width of such a decomposition is defined as $\max\{|X_i|\} - 1$. The *treewidth* of a graph is the minimum width of a tree decomposition of that graph. Note that trees have a treewidth of 1, hence the name.

2.2 Bipartite Graphs

A two-mode network, as opposed to one-mode networks (graphs), is a structure in which there are two types of objects, with links between two objects of different type, but not within the types. There are several ways to model this type of network. The first type is a bipartite graph. A *bipartite graph* is a graph where the set of vertices can be divided into two sets U and V such that all edges of the graph are between vertices of different sets.

Another way of modeling two-mode networks is through hypergraphs. A *hypergraph* is a generalization of graphs where edges are replaced by *hyperedges*, which are non-empty subsets of the set of vertices. Two vertices u and v are adjacent if there is a hyperedge that contains both u and v. The degree of a vertex v is equal to the number of hyperedges that contain v. Hypergraphs can also be seen as a *family of sets* over a universe.

This leads to the following problem, called Hitting Set. Given a universe U of size n and a collection of sets $S = \{S_1, \ldots, S_k\}$ with $S_i \subseteq U$, a hitting set is a set $C \subseteq U$ such that $\forall S_i \in S\colon S_i \cap C \neq \emptyset$. The problem Hitting Set in its optimization version asks for a hitting set of minimum cardinality. Another, similar problem is Set Cover. Given a universe U of size n and a collection of sets $S = \{S_1, \ldots, S_k\}$ with $S_i \subseteq U$, a set cover is a subset $C \subseteq S$ of the given sets such that $\bigcup C = U$. The problem Set Cover asks for a set cover of minimum cardinality. In fact, the two problems are identical: When treating the two-mode network as a bipartite graph, one problem can be converted to the other by exchanging the two types of vertices.

Two-mode networks can be converted to one-mode networks through a *projection*. This projection is lossy, since not all original information can be preserved. One such projection is the *intersection graph*.

Definition 2.1 (Intersection Graph)**.** *The intersection graph of a collection of sets $S = \{S_1, \ldots, S_k\}$ is defined as $G = (V, E)$ with $V = S$ and $E = \{\{S_i, S_j\} | S_i \cap S_j \neq \emptyset\}$.*

In other words, the intersection graph connects two sets if and only if they share an element. Again, there are two ways to look at this: from a two-mode

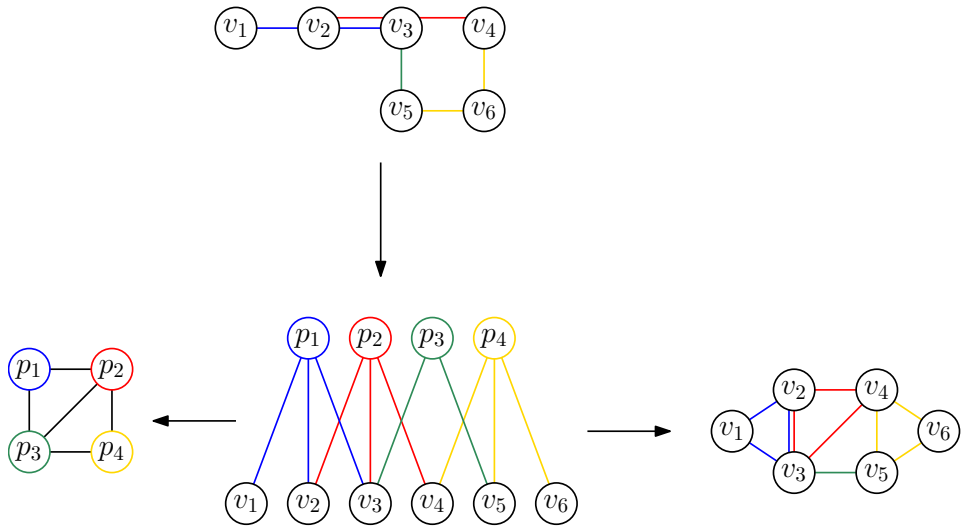


Figure 2.1: A Path Cover by Vertices instance (up), the corresponding Hitting Set instance as a bipartite graph (center), and two possible projections as intersection graphs based on the sets (on the left) and based on the vertices (on the right).

network, we can derive two intersection graphs, depending on the side we choose as the universe. Figure 2.1 shows the two possible intersection graphs of one bipartite graph.

2.3 Path Cover by Vertices

In this section, we introduce the Path Cover by Vertices problem and reduction rules. They were initially introduced by Weihe [38].

Definition 2.2 (Path Cover by Vertices; [38])**.** *Given an undirected graph $G = (V, E)$ and a set of paths P in G, such that, for each edge $e \in G$, there is a path $p \in P$ that uses the edge e. Let $P(v)$ denote the set of paths that include vertex v, and let $V(p)$ denote the set of vertices that are part of path p. The problem is to find a minimum path cover by vertices, that is, a subset of the vertices $C \subseteq V$ such that $\forall p \in P\colon V(p) \cap C \neq \emptyset$, so each path contains at least one chosen vertex.*

For this problem, we introduce the following notion of dominance. Let $v_1, v_2 \in V$. We say v_2 *dominates* v_1 if $P(v_1) \subseteq P(v_2)$. That means, every path that contains v_1 also contains v_2. Let $p_1, p_2 \in P$. We say p_1 *dominates* p_2 if $V(p_1) \subseteq V(p_2)$. This means, every vertex that would cover p_1 would also cover

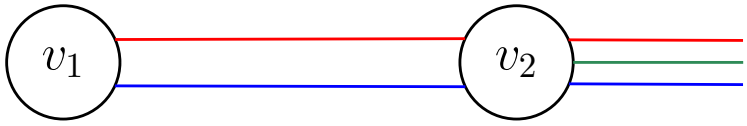


Figure 2.2: The vertex v_2 dominates v_1.

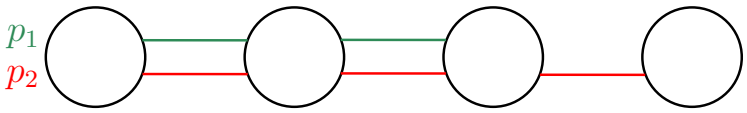


Figure 2.3: The path p_1 dominates p_2.

p_2. Note that, although the formulation is similar to vertex dominance, the roles of p_1 and p_2 are reversed here. Figure 2.2 illustrates dominance between vertices, while Figure 2.3 illustrates dominance between paths.

The following reduction rules can be defined and applied. They were initially stated by Weihe [38], but are given here again for completeness. A *vertex reduction* on a vertex $v \in V$ can be applied if v is dominated and works as follows:

- The vertex v and its incident edges are removed from G.
- For every path that contains v, v is removed from the path. If v was neither the start vertex nor the end vertex of the path, an edge connecting the predecessor and successor vertices is added to E if it is not contained already.

Since v is dominated, we can discard the vertex from our instance: for every path cover by vertices that would have included v, we can choose the dominating vertex instead of v.

Similarly, a *path reduction* on a path $p \in P$ can be applied if p is dominated and works as follows:

- The path p is removed from P.
- Every edge $e \in E$ that does not belong to any path is removed.

The path p is dominated by some other path p_1 with $V(p_1) \subseteq V(p)$. Since p_1 has to be covered in any path cover by vertices, the path p will be covered too. Therefore, we can discard the path p.

An *irreducible core* or simply *core* of a Path Cover by Vertices instance is created by repeatedly applying vertex reduction to a dominated vertex and

path reduction to a dominated path in an arbitrary order, until no reduction can be applied anymore. Evidently, every application of a reduction rule is computable in polynomial time, and every reduction reduces either the number of vertices or the number of paths.

In an irreducible core, we call vertices that are contained in only one path *isolated vertices*. These vertices cannot have any neighbors in G. The *isolated path* of an isolated vertex can only contain that vertex and no other vertices, otherwise the vertex would be dominated by another vertex.

We call an instance *solved* if all remaining vertices are isolated. The minimum path cover by vertices of a solved instance is the remaining set of vertices. An instance can be *solved by reduction* if its irreducible core is solved.

2.4 Fixed-Parameter Tractability

A problem of input size n is called *fixed-parameter tractable* in a parameter k if there is an algorithm with runtime complexity $f(k) \cdot n^{O(1)}$ for some computable function f. For a constant k, such an *FPT algorithm* allows for a polynomial run time that is independent of k (as opposed to a run time like $O(n^k)$). The class of fixed-parameter tractable problems is called FPT. For some problems and parameters, no FPT algorithm is known yet. These can be further categorized with the *W hierarchy*, which defines several problem classes $W[i]$. It is known that $\text{FPT} = W[0] \subseteq W[1] \subseteq W[2] \ldots$. The Independent Set problem with solution size k is known to be in $W[1]$. The Hitting Set problem with solution size k is in $W[2]$.

3 Parameterized Complexity

In this chapter, we focus on the parameterized complexity of the Path Cover by Vertices problem. Since a Path Cover by Vertices instance is on a graph G, the natural first step is to try to classify the effectiveness of the rules and the complexity of solving the problem based on the underlying graph.

3.1 Complexity based on the Underlying Graph

In order to understand the circumstances under which the reduction rules are the most effective, the first question we answer is:

> Which underlying graphs allow for Path Cover by Vertices instances to be solved by reduction?

The following lemmas answer that question: we show that this is the case for forests, but for any other graph, it is impossible to tell without examining the actual paths of the instance.

Lemma 3.1. *If G is a forest, the* Path Cover by Vertices *instance can be solved by reduction.*

Proof. A forest always has a leaf. Let v_l be this leaf and v_n its neighbor. Then it holds that either $P(v_l) \subseteq P(v_n)$ or $\exists p \in P(v_l)\colon p \notin P(v_n)$. In the first case, v_l is dominated by v_n, thus the leaf and its edge will be removed from the graph. In the second case, there is a path p that includes the leaf v_l, but not its neighbor v_n. Thus, $p = (v_l)$. The path p dominates all paths that use the edge $\{v_l, v_n\}$, thus the path reduction can be applied to all of these paths, eventually removing the edge. The leaf v_l will remain isolated for the remainder of the reduction. The remaining graph is still a forest but with at

least one edge removed. Thus, by induction over the leaves, there is always one reduction rule applicable until only isolated vertices remain. □

Now that we have shown that the reduction rules are very effective for forests, we examine the opposite case: we show that if the graph contains a cycle, we cannot tell based on the graph alone whether the instance can be solved by reduction.

Lemma 3.2. *For any graph G that contains a cycle, there is a* Path Cover by Vertices *instance on G that can not be solved by reduction.*

Proof. For the proof, we choose an arbitrary orientation $u < v$ for every edge $\{u, v\} \in E$. For the graph $G = (V, E)$, fix an arbitrary cycle of vertices V_c and set $P = \{(u, v) \mid \{u, v\} \in E \wedge u < v\} \cup \{(v) \mid v \in V \setminus V_C\}$, such that there is a path for each edge as well as a path for each vertex that is not in the fixed cycle. Then, all the edge-paths (u, v) not part of the cycle will be dominated by the vertex-path (u) or (v), since one of them is not in the cycle. Thus, all of these edge-paths and their edges will be removed. All vertices not in the cycle are isolated and cannot be part of any other dominations. For the remaining cycle, no further reduction is possible. There is no path dominance because all paths in the cycle contain two elements, and no two paths contain the same two elements. There can be no vertex dominance either since all vertices in the cycle are part of two paths, and these two paths cannot be the same for two different vertices. Since no further reductions can be applied, the instance can not be solved by reduction. □

However, with the right paths, even graphs that contain a cycle can be solved by reduction, as we show in the following Lemma.

Lemma 3.3. *For any graph G, there is a* Path Cover by Vertices *instance on G that can be solved by reduction.*

Proof. For the proof, we choose an arbitrary orientation $u < v$ for every edge $\{u, v\} \in E$. For the graph $G = (V, E)$, set $P = \{(u, v) \mid \{u, v\} \in E \wedge u < v\} \cup \{(v) \mid v \in V\}$, such that there is a path for each edge as well as a path for each vertex of the graph. Then, each edge-path (u, v) will be dominated by the vertex-path (u). All edge-paths and all edges will be removed, leaving only the vertex-paths. All remaining vertices are isolated, thus the instance is solved. □

These lemmas in combination with Lemma 3.1 yield the following corollary.

Corollary 3.4. *Any* Path Cover by Vertices *instance where G is a forest can be completely solved by the reduction rules. For any other graph G, either case is possible.*

This means that, when only looking at the graph G, this classification is the best possible. Thus, it completely answers which graphs allow for an instance to be solved by reduction. But maybe the requirement of being solved by reduction is too tight. Instead, we generalize this question and ask for graph classes that allow for a polynomial time algorithm.

> Which instance classes of Path Cover by Vertices can be solved in polynomial time?

Since we have shown in Lemma 3.1 that all forests can be solved by reduction, and the reduction runs in polynomial time, the following Corollary follows directly.

Corollary 3.5. *The class of* Path Cover by Vertices *instances on forests can be solved in polynomial time.*

The following lemma deepens the insights of Lemma 3.2. We show that, even though instances with cycles might not always be solvable by reduction, instances on a cycle graph (a graph that consists of a single cycle) can nonetheless be solved in polynomial time.

Lemma 3.6. Path Cover by Vertices *instances on cycle graphs can be solved in polynomial time.*

Proof. We prove this by giving a polynomial-time algorithm. If there is at least one path to cover, the solution vertex-set is non-empty. We branch on the decision which vertex to include in the cover. Suppose, in this branch, $v \in V$ has been chosen. We can remove v, its occurrences in paths and its edges, such that a cycle-free graph remains. This can be solved in polynomial time, as shown above. Since there are $|V|$ vertices to branch on, this results in a polynomial-time algorithm. □

The previous results suggest that graphs that are tree-like might be easy to solve. Trees can be solved by reduction, and cycles can be broken with some extra effort. This raises the question whether a parameterization by tree-likeness or some other parameter can aid at solving the Path Cover by Vertices problem.

> Are there FPT algorithms for Path Cover by Vertices on parameters of the graph of the instance?

We show that parameterizations by several variants of tree-likeness, namely treewidth and feedback vertex number, are impossible unless P=NP. The proof for the following lemma is based on a proof by Jansen [25], who shows NP-completeness of Path Cover by Vertices in the feedback vertex number 2 based on a reduction from a problem called n-Totally Ordered Regular Signed 3-SAT. We simplify the proof and give a detailed explanation for both feedback vertex number and treewidth.

Lemma 3.7. Path Cover by Vertices *is NP-hard for graphs of treewidth 3 and for graphs with the feedback vertex number 2.*

Proof. We prove this by reducing from 3-SAT. Let φ be a propositional formula on n variables. We create two vertices x and $\overline{x}$ for every literal and connect them. Then, we add two vertices z_0 and z_1 which are connected to every x vertex. For every literal, we add a path $(x, \overline{x})$. For every clause $(a \vee b \vee c)$ we add a path $(x_a, z_0, x_b, z_1, x_c)$ where each x_i is the vertex $\overline{x_i}$ or x_i based on whether the variable is inverted or not. Figure 3.1 illustrates this reduction.

Now, if there is a path cover by vertices of size n (where n is the number of variables), then the SAT instance is solvable. Since for every variable x, the vertex x or $\overline{x}$ has to be chosen, and there are n such variables, exactly one of the two vertices x and $\overline{x}$ is chosen. Thus, neither z_0 nor z_1 can be part of the path cover by vertices. The correct variable assignment is given by setting a variable x to false if $\overline{x}$ is chosen and true otherwise. Then, every clause is fulfilled, since the path for that clause is covered by at least one vertex belonging to a literal that is chosen. Similarly, every valid variable assignment yields a path cover by vertices of size n. For every variable x, we choose the vertex x if x is set to true and choose $\overline{x}$ otherwise. Thus, each path $(x, \overline{x})$ is covered. Additionally, all the paths corresponding to clauses are covered, since either of the three literals was true in the variable assignment. This way, we get a path cover by vertices of size n.

The graph created has a feedback vertex number of 2: if we remove both z_0 and z_1 from the graph, the only edges left are those between the vertices x and $\overline{x}$ for each variable. The graph also has treewidth 3. Consider the following tree decomposition. For every variable x, we add a bag $\{x, \overline{x}, z_0, z_1\}$. Our tree decomposition now consists of all of these bags in one single path in an arbitrary order. Each vertex is in some bag. For every edge, there is a bag that contains both of its vertices. Finally, the only vertices that occur in multiple bags are z_0 and z_1, and since they are in every bag, this forms a subtree. Each bag has a size of 4, thus the graph has a treewidth of 3. □

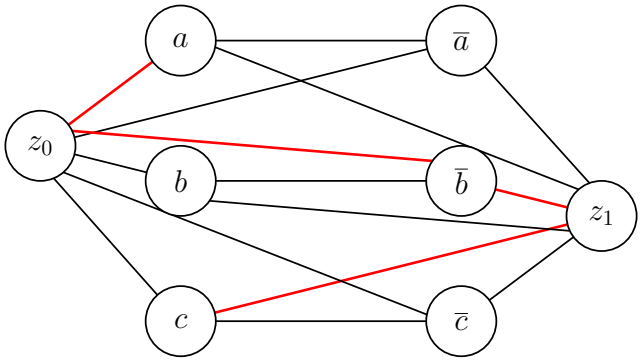


Figure 3.1: Example for the reduction from 3-SAT to Path Cover by Vertices with treewidth 3. The path shown in red is the path that is created for the clause $a \vee \overline{b} \vee c$.

This result is surprising in the following sense. By Courcelle's Theorem, most graph-related problems admit FPT algorithms when parameterized in treewidth [13]. However, we have shown that the problem is already hard for constant treewidth. That means, assuming P≠NP does not only rule out a polynomial algorithm for PVC, but also any FPT algorithm [18]. We have shown that the problem is polynomial time solvable on trees (treewidth 1) and NP-hard for treewidth at least 3. It remains open whether the problem is NP-hard for graphs of treewidth 2.

Another hardness result comes from the problem's relation to Vertex Cover.

Lemma 3.8. *If* Vertex Cover *is NP-hard on a graph class, then* Path Cover by Vertices *is NP-hard on that graph class too.*

Proof. We prove this by reducing from arbitrary Vertex Cover instances to Path Cover by Vertices instances on the same graph. Given a Vertex Cover instance in the form of graph $G = (V, E)$, we choose an arbitrary orientation $u < v$ for every edge $\{u, v\} \in E$. We create the Path Cover by Vertices instance on the same graph G with the paths $P = \{(u, v) \mid \{u, v\} \in E \wedge u < v\}$. This yields a valid Path Cover by Vertices instance, since every edge has a corresponding path. Clearly, any vertex cover corresponds to the same path cover by vertices and vice versa. □

Based on the reduction to Vertex Cover in Lemma 3.8, it follows that Path Cover by Vertices is already difficult to solve even for very short paths.

Corollary 3.9. Path Cover by Vertices *remains NP-hard if each path has length 1.*

Garey et al. [20] show that Vertex Cover remains NP-hard on planar graphs with vertex degree at most 3. By Lemma 3.8, this leads to the following corollary.

Corollary 3.10. Path Cover by Vertices *remains NP-hard on planar graphs with vertex degree at most 3.*

All of these results on NP-hardness are very strong indicators that the effectiveness of the reduction rules cannot be derived from the graph structure of a Path Cover by Vertices instance. Instead, we now look at parameters that do not utilize the graph structure of the instance. This is equivalent to the Hitting Set problem.

3.2 Complexity based on Hitting Set

Any Path Cover by Vertices instance can be converted to a Hitting Set instance by using the set of vertices as the universe and creating a subset for each path containing its vertices. Additionally, the reduction rules immediately translate into corresponding reduction rules for Hitting Set instances, since the rules do not take the order of elements in the path into account. An element e_2 dominates an element e_1 if $S(e_1) \subseteq S(e_2)$, where $S(e)$ denotes the collection of subsets that contain e. In the *element reduction* the dominated element is removed from the universe and from all sets containing it. A set s_1 dominates s_2 if $s_1 \subseteq s_2$. In the *set reduction* the dominated set is removed from the family of sets. The *core* term applies accordingly. This conversion allows to further investigate the complexity of solving Path Cover by Vertices. We give an overview of the research on the Hitting Set problem and describe their relation to our Path Cover by Vertices problem. The reduction rules were described by Nemhauser and Wolsey [30]. The NP-hardness of Hitting Set was shown by Karp [27]. Additionally, it was shown by Downey and Fellows that the problem is W[2]-complete in the solution size [15], meaning the problem is not tractable with respect to the solution size unless W[2]=FPT. A well-known result is that Hitting Set is fixed-parameter tractable in the solution size k and a maximal set size d. The Hitting Set problem for a maximal set size d is called d-Hitting Set. The 2-Hitting Set problem is equivalent to the Vertex Cover problem, which is well-researched from an FPT perspective, achieving a runtime of $1.2738^k n^{O(1)}$ [10]. Similarly, the 3-Hitting Set problem is well-researched [31, 1]. However, such a parameterization in the maximal set size d does not

explain the effectiveness of the reduction rules in real-world instances, since transit networks often contain lines that cover up to 100 stations, as we show in Chapter 4 and Chapter 5. Bringmann et al. showed that Hitting Set is W[1]-hard when the instance has a low VC-dimension [9]. A low VC-dimension means that the largest subset of the universe which appears in all variations in the sets has few elements.

The Hitting Set problem is known to be equivalent to the Set Cover problem [4]. A common approach to finding suitable parameterizations and attributes of Hitting Set and Set Cover instances is to look at the binary matrix representing the instance. For a Hitting Set instance, each row of the matrix represents a set that should be hit, with a 1 in every column that corresponds to an element of the set. A matrix has the *strong consecutive ones property* if, in every row, all ones are consecutive [14]. A matrix has the *consecutive ones property* if there is a permutation of the rows such that the resulting matrix has the strong consecutive ones property. Similarly, a matrix has the *strong circular ones property* if, in every row, all ones are consecutive or all zeros are consecutive. The *circular ones property* is defined accordingly. This allows for the ones to "wrap over the edge" of the matrix. Both for the consecutive ones property and the circular ones property, there are efficient algorithms known to test whether a given matrix has this property by finding the correct permutation of the columns [7, 23]. Hitting Set instances that fulfill the consecutive ones property or the circular ones property can be solved in polynomial time [35]. There is a close relation to the Path Cover by Vertices problem: any instance on a path graph or a cycle graph has a corresponding Hitting Set instance with the consecutive ones property or the circular ones property respectively, by using the vertex order of the graph as the column order in the matrix. Ruf and Schöbel investigated these properties further by introducing the *almost consecutive ones property* [34]. A matrix has this property when the number of blocks of ones is very small. The authors show that this property is very useful for finding bounds and developing a branching algorithm for the Hitting Set problem.

In the context of the Set Cover problem, Guo and Niedermeier introduce tree-like set systems [21]. In a tree-like set system of subsets C over a base set S, there is a tree T such that each set from the set system corresponds to a vertex in the tree, and for any $s \in S$, the vertices of the subsets that contain s induce a subtree. In the context of Hitting Set, this is very similar to our Path Cover by Vertices problem, with the modification that the underlying graph G is a tree and we do not consider paths in G, but subtrees.

Jansen parameterizes the distance to a tree-like set system via the cyclomatic number, the feedback vertex number and the treewidth of the underlying graph G [25]. While an FPT algorithm in the cyclomatic number is found, the author shows that a parameterization in the feedback vertex number or the treewidth is not possible unless P=NP.

The following result is based on a proposition given by Weihe [38].
Lemma 3.11. *For general graphs,* Path Cover by Vertices *is W[2]-hard in the solution size.*

Proof. We prove this by reducing an arbitrary Hitting Set instance to a Path Cover by Vertices instance. This is done by putting all elements of a set into an arbitrary order and using this order for a path in the graph. The solution size remains unchanged: each path corresponds to a set, so any hitting set corresponds to a path cover by vertices of the same size. Since Hitting Set is W[2]-hard in the solution size [15], so is Path Cover by Vertices. □

A Path Cover by Vertices instance can be converted to a corresponding Hitting Set instance, but a given Hitting Set instance has several possible Path Cover by Vertices instances since the elements of each set can be put into a path in an arbitrary order, yielding different graphs. One task that comes up is the task of finding a suitable representation of a given Hitting Set instance that allows for easy solving. For example, if there is a way to represent a given Hitting Set with a Path Cover by Vertices instance that is a forest or cycle, the Hitting Set instance can be solved in polynomial time (Lemma 3.1 and Lemma 3.6).

Checking whether a Hitting Set instance has a Path Cover by Vertices instance on a cycle graph can be done with the circular ones property, as mentioned above. This means all elements can be arranged in a cycle graph such that all sets in the set family only contain consecutive elements in the cycle graph. The question arises whether a favorable Path Cover by Vertices representation for a given Hitting Set instance can be calculated efficiently. The following lemma shows that this is NP-hard when trying to minimize the number of edges in the graph.
Lemma 3.12. *It is NP-hard to find a* Path Cover by Vertices *instance of a* Hitting Set *instance with a minimum number of edges.*

Proof. We prove this by reducing an arbitrary Vertex Cover instance to this problem. We are given the graph $G = (V, E)$. From this, we create

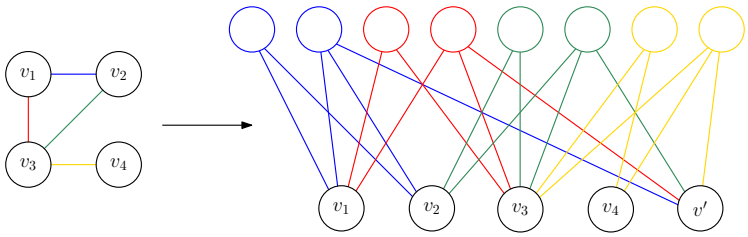


Figure 3.2: An example for the reduction of a Vertex Cover instance to the problem of finding a Path Cover by Vertices instance with a minimum number of edges. For each edge, we create one set containing both vertices and one set that additionally contains v'.

a Hitting Set instance as follows. For our universe, we use the set of vertices, but add a new vertex v', so $U = V \cup \{v'\}$. The sets to hit are $\{\{u, v\} \mid \{u, v\} \in E\} \cup \{\{u, v, v'\} \mid \{u, v\} \in E\}$. Evidently, the Hitting Set instance has the same minimum solution size as the Vertex Cover instance. Now we ask for a Path Cover by Vertices instance on a graph G' of this Hitting Set instance with a minimum number of edges. Figure 3.2 illustrates this reduction. All edges in G have to be in the graph G' of the Path Cover by Vertices instance, and for each edge $\{u, v\}$, u or v needs to be connected to v'. The neighbors of v' form a vertex cover: for each edge, at least one adjacent vertex is connected to v'. Furthermore, the size of the neighborhood has to be minimal, because we ask for a minimum number of edges. Therefore, we could derive a minimum vertex cover from the neighborhood of v'. Since Vertex Cover is NP-hard, the proof is complete. □

4 Average-Case

From the previous chapter, we derive two findings. We found that the Path Cover by Vertices problem is a Hitting Set problem in disguise, and the reduction rules are applicable for both problems. Therefore, we focus on the analysis of the effect of the reduction rules on Hitting Set instances. Furthermore, our parameterized complexity analysis did not yield any results that might explain the effectiveness of the reduction rules. This analysis focuses on often unrealistic worst cases. As opposed to that, there is also the technique of the average-case analysis, in which we investigate the effect of an algorithm (in our case the reduction rules) on typical instances. As noted by Karp, this approach can yield explanations that would have been hard to infer from a worst-case analysis [26]. Such an average-case analysis requires a probability distribution over the possible inputs. Choosing a realistic distribution is crucial to the meaningfulness the result of the analysis, but the more complicated the model for the distribution is, the harder it is to analyze. Therefore, we want to focus on key properties of real-world instances.

In order to develop a model for typical instances, we first analyze real-world transit networks. We hypothesize that stations are very heterogeneous, meaning there are few central stations with many lines going through them and many less important stations with only few lines. In the context of Hitting Set, for any element e from the universe, we call the number of sets that contain e the *degree* of e. Furthermore, we think that real-world transit networks are very clustered, meaning that two stations that both share a line with a third station are more probable to share a line than two randomly chosen stations. We will give a formal definition of this clustering in Section 4.1. We conjecture the following claims for the Hitting Set problem.

- Real-world instances are heterogeneous and clustered.
- The reduction rules are more effective the more heterogeneous the ele-

ment degrees are.

- The reduction rules are more effective the more clustered the instances are.

The heterogeneity of an instance can be checked by creating histograms of the element degrees of the instances. To measure the clustering, one first has to agree on a suitable generalization of the clustering coefficient to Hitting Set instances (two-mode networks). Therefore, Section 4.1 focuses on several approaches towards measuring the clustering of a given instance. Section 4.2 describes the measured parameters for real-world data. Section 4.3 then describes how we generate random instances that have the desired properties of heterogeneity and clustering with scalable parameters. Finally, Section 4.4 shows the results for the generated instances, which provides some context for the real-world transit networks.

The focus of the analysis lies on measuring the size of the instance after application of the reduction rules. This size can be described with the *relative core size*. For a Hitting Set instance with universe U and sets S, the *relative core size* describes the ratio

$$\frac{\sum_{S'_i \in S'} |S'_i| - |\{S'_i \mid |S'_i| = 1\}|}{\sum_{S_i \in S} |S_i|},$$

where S' is the set collection of the core. Note that the term $|\{S'_i \mid |S'_i| = 1\}|$ is the number of sets of size 1, so we account for partial solutions found by the reduction. A relative core size of 0 means the instance was completely solved, while a relative core size of 1 means the instance could not be reduced at all.

4.1 Clustering Parameters

In order to measure the clustering of a Hitting Set instance, we can examine the underlying bipartite graph. The field of measuring the clustering of a network is well-researched, although much of it was focused on the analysis of general graphs as opposed to bipartite graphs [37, 2]. One prominent parameter for such one-mode graphs is the *clustering coefficient*, which can be found in several variants [36]. We focus on the following variant, often called the *global clustering coefficient*. The global clustering coefficient of a graph models the probability that, given three vertices with at least two edges be-

tween them, the third edge is present too. This is what one would expect e.g. in social networks: it should be probable that two of someone's friends are friends with each other, too. More formally, the global clustering coefficient of a graph is the ratio of the number of triangles to the number of unordered connected triplets of vertices.

This approach does not work in bipartite graphs, since there can be no cycles of odd length. Instead, we compare several ways of measuring clustering in two-mode networks. The first definition we examine is that given by Robins and Alexander [33]. It generalizes the clustering coefficient to cycles of length 4. It is defined as the ratio between the number of cycles of length 4 and the number of paths of length 3. We call this parameter C_4.

Another approach is to use a projection of the bipartite graph, such as an intersection graph. Then, we can measure the clustering coefficient of the projection. Recall that the intersection graph of a family of sets is formed by creating a vertex for each set and connecting two sets with an edge if and only if they share an element. Given a Hitting Set instance, we can create two intersection graphs: One uses the family of sets as described above, the other is based on the elements and connects two elements if there is a set that contains both. We call the clustering coefficient of the intersection graph of the set of elements C_E, while the clustering coefficient of the intersection graph of the family of sets will be called C_S.

The projection does not contain the full information; if three sets form a triangle in the projection, we do not know whether this connection is due to one element which is part of all three sets or whether there are three different elements that are part of two of the sets each (see Figure 4.1). Therefore, we introduce another parameter $P_\triangle$. It is defined based on the projection using the family of sets. The parameter measures the probability for a triangle in the intersection graph that the three participating sets share an element (as opposed to being only pairwise connected). More formally, we define $P_\triangle$ as

$$\frac{\left|\left\{\{A,B,C\} \in \binom{S}{3} \mid A \cap B \cap C \neq \emptyset\right\}\right|}{\left|\left\{\{A,B,C\} \in \binom{S}{3} \mid A \cap B, A \cap C, B \cap C \neq \emptyset\right\}\right|}.$$

We expect this parameter to be relatively high for real-world instances and a good measure for clustering. With respect to the Path Cover by Vertices problem, a high parameter tells us that three paths sharing an element and therefore being close to each other is much more probable than three paths forming a triangle along great distances.

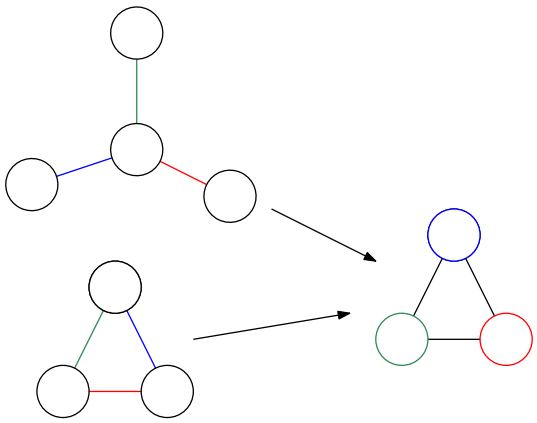


Figure 4.1: Lost information during projection: both Path Cover by Vertices instances on the left yield the same intersection graph of sets.

4.2 Analysis of Real-World Data

In a first step, we replicate the results of Weihe [38] on the effectiveness of the reduction rules by using publicly available transit data from several countries. We arrive at very similar results and test our hypothesis on the heterogeneity of the instances. The results on the heterogeneity form the foundation of our instance generation.

The data was extracted from source files in the GTFS format. In this data format, there are several paths for a similar route — one for each time a vehicle actually drives the route. For each route, only the first path was taken, and paths that cover the same set of stations were ignored.

The first dataset comes from the Netherlands and contains transit data from buses, metro, trains, trams and some ferries across all of the Netherlands[1]. The second dataset comes from VBB (Verkehrsbund Berlin-Brandenburg) and contains transit data around Germany's capital, including buses, trams and trains[2]. The third dataset comes from France and contains the transit data of all the regional TER trains of the SNCF (Société nationale des chemins de fer français)[3]. The fourth dataset comes from DB (Deutsche Bahn) and contains the transit data of long-distance trains Germany[4]. The names of these four

1 Available at `https://old.datahub.io/dataset/gtfs-nl`
2 Available at `http://daten.berlin.de/kategorie/verkehr`
3 Available at `https://ressources.data.sncf.com/explore/dataset/sncf-ter-gtfs/information/`
4 Available at `http://data.deutschebahn.com/dataset/api-fahrplan`

Dataset	Vertex count	Path count	Relative core size
NL	31250	2804	0.019
VBB	13424	1241	0.016
SNCF	3789	974	0.005
DB	514	586	0.000

Table 4.1: Overview of the datasets, their sizes and their relative core size. All four datasets yield a very small core.

datasets will be shortened to NL, VBB, SNCF and DB, respectively.

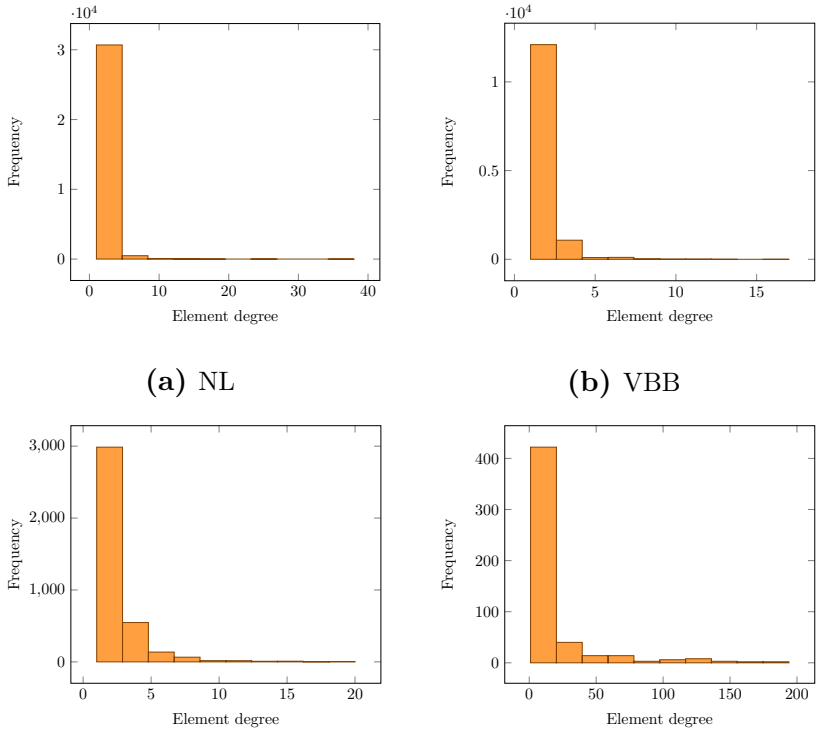
Table 4.1 gives an overview of the different data sources and their size. The results are in line with the original paper's findings, leaving only a small fraction of the original instance size as the core.

4.2.1 Element degree distribution

Figure 4.2 shows the histogram of the element degrees for the real-world instances, with each element corresponding to a station. This shows already very clearly that the element degrees are heterogeneous. To further manifest this observation, Figure 4.3 shows the complementary cumulative distribution function (CCDF) in a log-log plot. For a value x, the CCDF of a variable describes the probability that the variable will have a value less than or equal to x. In our case, this variable is the degree of a certain element.

The linearity in the log-log plot indicates that this is close to a power-law distribution. In the context of the Hitting Set problem, a power-law distribution of the element degrees means that the number of elements which are part of at most x sets is proportional to $x^{-\beta}$ for some constant β. Estimating the power-law exponent β with the python package `powerlaw` [3] yields $\beta \approx 4.0$ for NL, $\beta \approx 4.1$ for VBB, $\beta \approx 3.0$ for SNCF, and $\beta \approx 2.0$ for DB. The corresponding estimated distribution for each dataset can be seen in Figure 4.3. It should be noted that the DB dataset somewhat represents an outlier. For one, the instance is much smaller than the other datasets. Second, it contains elements with a degree of nearly 200, which is why the element degree distribution does not seem to be power-law distributed.

For the other instances, the power-law distribution can be explained as follows. The element degree is correlated to the importance of the station (a central station has the most connections going through it), and the sizes of cities are



(a) NL **(b)** VBB

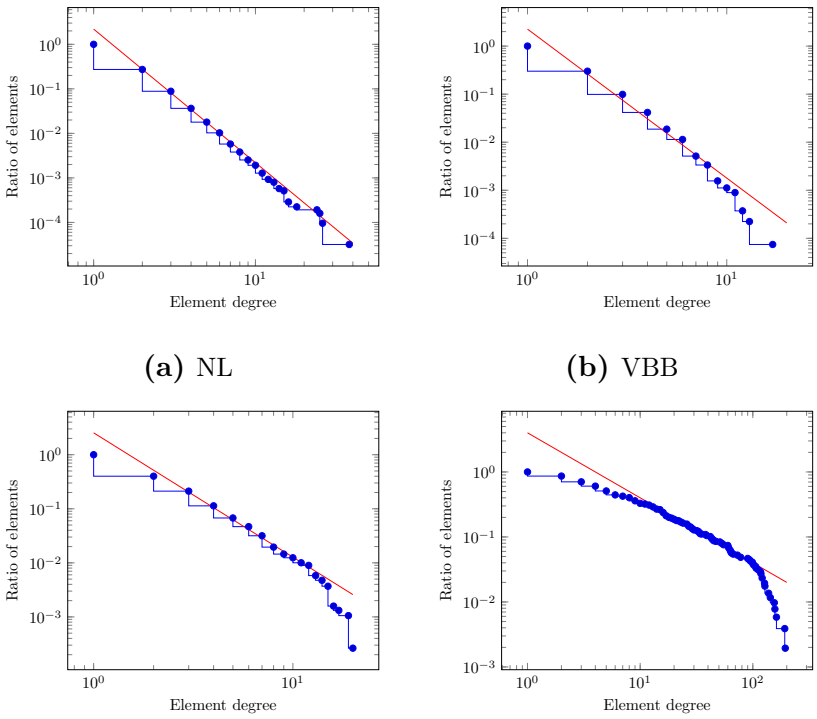
(c) SNCF **(d)** DB

Figure 4.2: Histograms of element degrees for real-world data.

known to be distributed according to a power-law distribution [19]. This reinforces the suggestion that real-world instances exhibit heterogeneous element degrees.

4.2.2 Set size distribution

Figure 4.4 shows the histograms of the set sizes for the real-world data. The set sizes in our datasets are not particularly heterogeneous. Again, this seems to match expectations since transit lines are designed rather than evolved: lines that cover longer distances do not stop at more stations, they often even tend to stop less frequent. Different types of public transport all cover about the same number of stations.



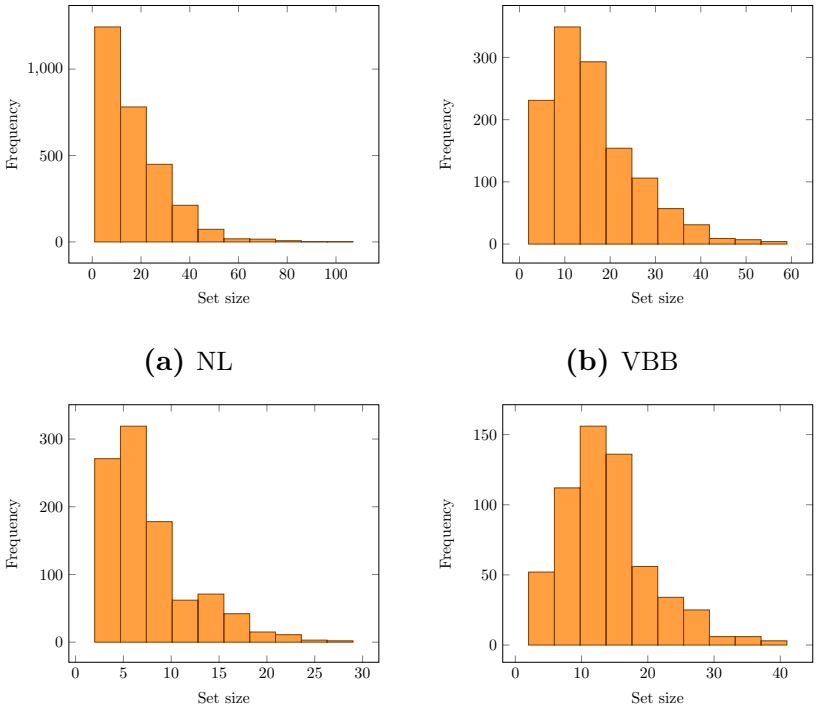
(a) NL **(b)** VBB

(c) SNCF **(d)** DB

Figure 4.3: The CCDF of element degrees for real-world data. The blue line shows the respective dataset, while the red line shows the estimated power-law distribution.

4.3 Instance Generation

Based on the findings in real-world instances described above, we want to generate random instances that are both heterogeneous and clustered. The random instances of Hitting Set are generated with the following approach for generating bipartite graphs, which is based on the model of geometric inhomogeneous random graphs (GIRGs) by Bringmann et al. [8]. As input, the model receives the number of desired elements and sets. All elements and sets are given a weight according to some weight function w_s and w_e. The expected degree of an element will be proportional to the element's weight. Due to the results on heterogeneity in Subsection 4.2.2 and Subsection 4.2.1, the sets receive weights according to a uniform distribution, while the element weights are distributed according to a power law with exponent β. The variables W_s and W_e are the total sum of the set weights and element weights. Each element



(a) NL **(b)** VBB

(c) SNCF **(d)** DB

Figure 4.4: Histograms of set sizes for real-world data.

e and each set s is represented as a random point on a circle, distributed uniformly. The distance $\text{dist}(s, e)$ of a set s and an element e is defined as the distance of the points on the circle. We use this distance to have control over the clustering of our resulting instance.

Membership of some element e in a set s is represented by the edge (s, e). Each such edge is sampled with probability

$$P(s, e) = \min\left(1, c \cdot \left(\frac{w_s(s)w_e(e)}{W_s W_e \text{dist}(s, e)}\right)^{1/T}\right).$$

The temperature T, which is between 0 and 1, controls the influence of the distance function and the weights. A low temperature corresponds to high heterogeneity, and as $T \to \infty$, the graph is more homogeneous. Constant c is used to scale the number of edges in the created bipartite graph. The power-law distribution of the element weights can be controlled with the parameter

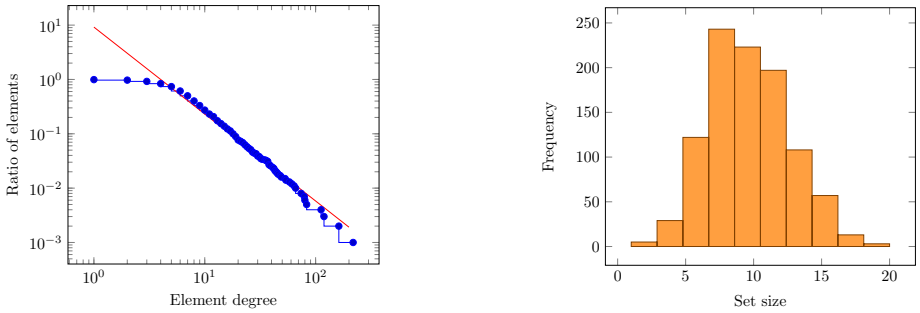


Figure 4.5: The CCDF of the element degrees and the set size distribution for a generated instance with $\beta = 2.5$ and $T = 0.9$.

β. The lower β is, the more heterogeneous the instances are. Note that, as $\beta \to \infty$, a uniform element weight distribution is approached. Figure 4.5 shows the distribution of element degrees in CCDF in a log-log plot and the distribution of set sizes for a generated instance with $\beta = 2.5$ and $T = 0.9$. Just as desired, the element degrees are power-law distributed, while the set sizes show a more homogeneous distribution similar to those of the real-world datasets.

4.4 Results

The reduction rules and the graph generation were implemented in C++. Note that the exact specifications of the machine are irrelevant since we do not measure any run times.

First, we investigate the relative core size in dependence of the two parameters of clustering (the temperature T) and the heterogeneity (the power-law exponent β) in combination. For this experiment, graphs with 1000 elements, 1000 sets, 10000 edges, a varying temperature T between 0 and 1 (in steps of 0.05) and a varying power-law exponent between 2 and 5 (in steps of 0.25) were generated. For each data point, we take 10 samples and calculate the mean of the relative core size. Figure 4.6 shows the results of the measurements. Small power-law exponents as well as temperatures have a positive impact on the effectiveness of the reduction rules for small values, although the clustering seems to be more vital. As the temperature goes to 1, the impact of the power-law exponent is greatly reduced. On the other hand, a small temperature leads to a small relative core size, even as the element weight distribution

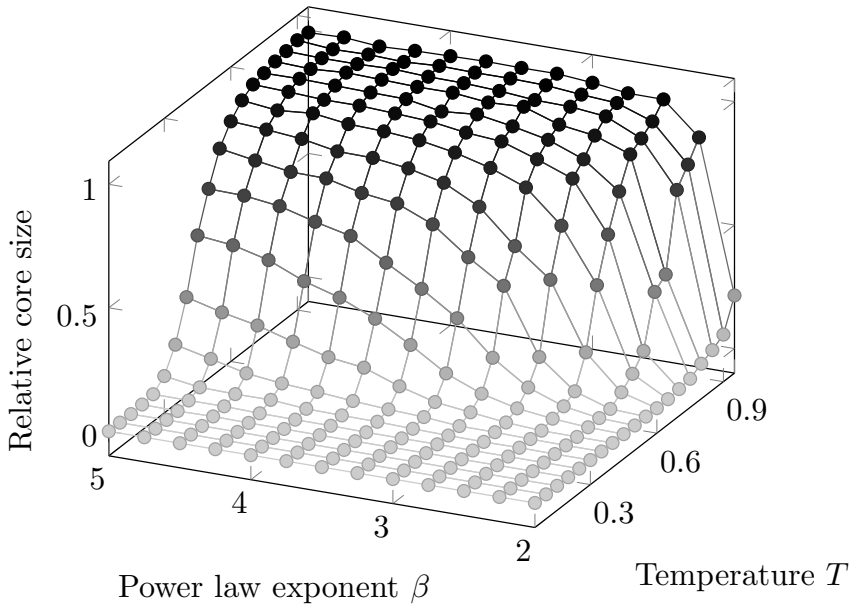


Figure 4.6: The relative core size for varying values of β and T. A low temperature strongly correlates with a small relative core size.

goes towards a uniform distribution.

To better understand this behavior and the relation to the measured parameters, we present further experiments that focus on the effect of the temperature on the relative core size for fixed values of β. For these experiments, graphs with 1000 elements, 1000 sets, 10000 edges, a varying temperature T between 0 and 1 (in steps of 0.02) and several values of β were generated. For each data point, we take 10 samples and calculate the mean of the measured attributes (relative core size and parameters). Figure 4.7 shows the relation between the temperature T and the relative core size for a power-law exponent for different values of β. Note that $\beta = \infty$ is equivalent to a uniform element weight distribution. This manifests the previous observation that a strong clustering leads to a small relative core size, even if the instances are homogeneous.

Figure 4.8 shows the relation between the temperature T and the clustering parameters for different levels of heterogeneity. For each parameter, a low value corresponds to a high temperature and therefore low clustering, while a high value corresponds to a low temperature and therefore high clustering. Thus, all parameters are appropriate measures for the clustering of the instances.

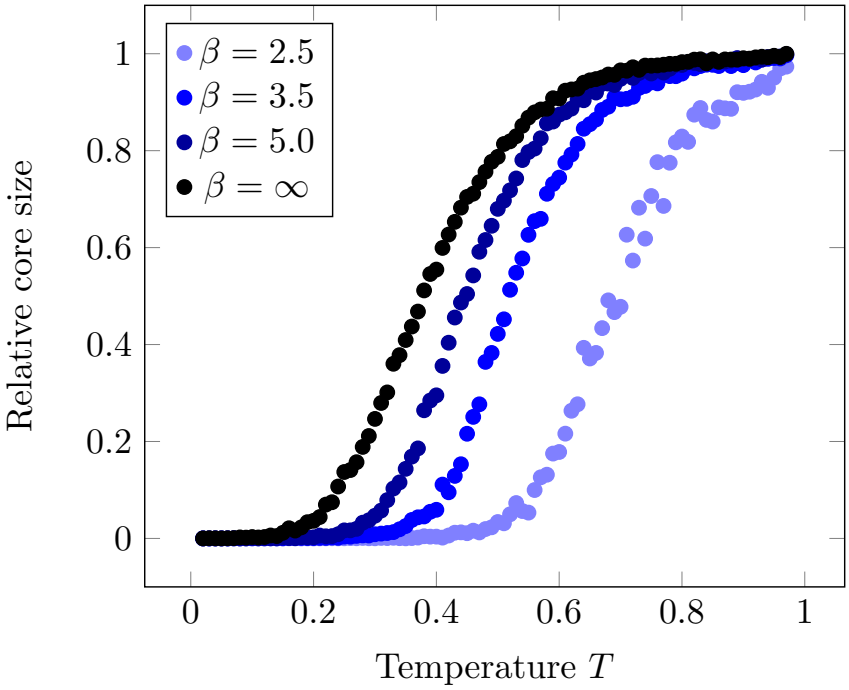


Figure 4.7: Relation between the temperature T and the relative core size for various degrees of heterogeneity.

However, while the clustering parameters are supposed to be correlated with the clustering of the instance, they also show a dependence on the heterogeneity. Note that both C_4 and C_E show very different values for different levels of heterogeneity. We offer an explanation for this behavior. Recall that C_4 models the probability that, for a path of length 3 in the bipartite graph, the edge between the first and last element/set in this path is present as well. A common structure found in real-world transit networks is that of a central station that has several lines going through it that share no other stations (e.g., because these lines spread to different directions). This is shown in Figure 4.9.

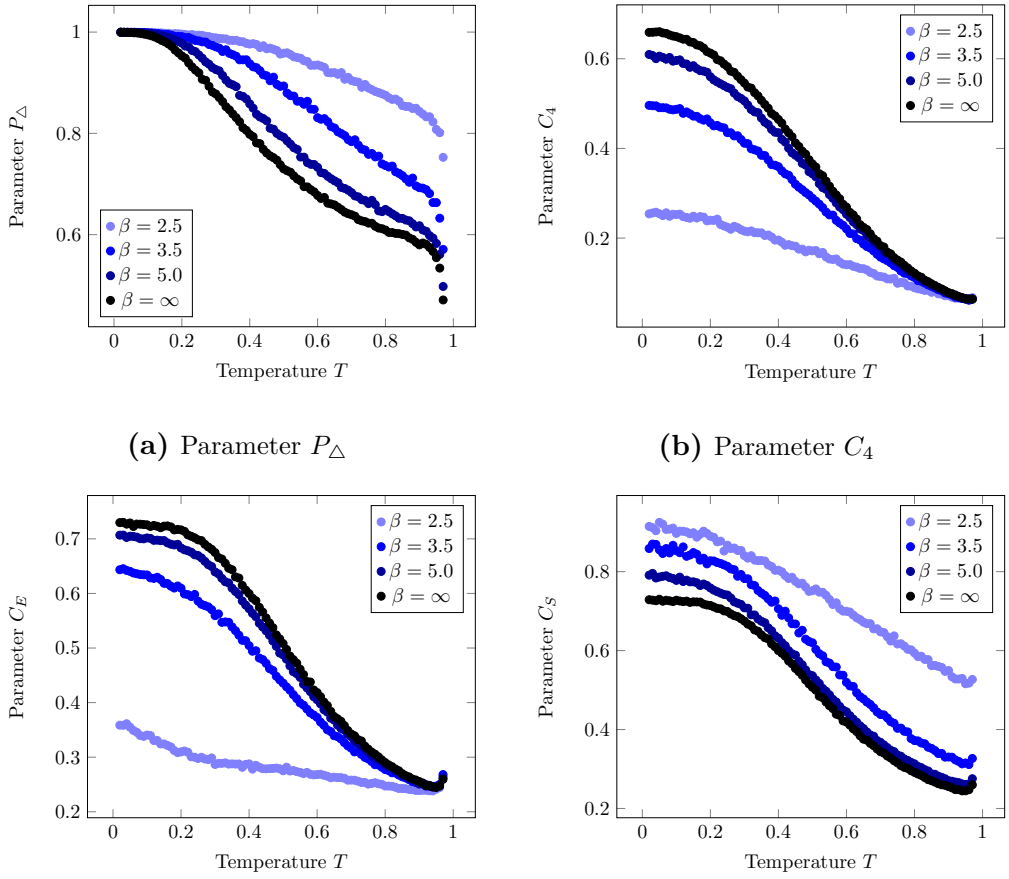
The bipartite graph of this structure contains many paths of length 3. These paths start in an arbitrary set, run through the element representing the central station, through a different set to another element. This first set and the last element are not connected, since the element is not part of the set. This leads to a small C_4 value. Similarly, each pair of elements from different sets forms an uncompleted triangle that lowers the C_E parameter. This kind of structure

is common in real-world networks, and it is more present in the generated instances the more heterogeneous the element weights are. This explains why both of these parameters are much smaller for higher values of β. Similarly, we can explain the behavior for the parameter $P_\triangle$. Any element with high degree increases the number of triplets of sets that share an element. Therefore, more heterogeneity increases the parameter $P_\triangle$.

Figure 4.10 shows the relation between the examined parameters and the relative core size. We can see that a large parameter corresponds to a small relative core size, while a small parameter corresponds to a large relative core size, meaning almost no reduction rules were applied. This confirms what should follow from the previous findings. The clustering parameters are good measures for the clustering, and high clustering yields a small relative core size. Therefore, a high parameter should indicate a small relative core size. With respect to their correlation to the relative core size, both the $P_\triangle$ and the C_S parameter are more resilient to the influence of the heterogeneity of the instances. The parameters C_4 and C_E on the other hand correspond to a very varying relative core size for different levels of heterogeneity. Again, this can be explained by the same observation that high heterogeneity has a large influence on both of these parameters.

We also compare the results of our model with the measurements for the real-world instances, which are marked in red. This comparison however should be taken with a grain of salt due to the small sample size of only four datasets. Nonetheless, the measured clustering parameters of these real-world instances are very high, and their relative core size is within the range of what our model predicts. Note that for the parameters $P_\triangle$ and C_E, the values for DB instance are outliers. We believe this is because the DB instance itself is an outlier with respect to the instance size and the heterogeneity, as discussed above.

In this chapter, we have analyzed real-world transit network instances of the Hitting Set problem and shown that they are clustered and heterogeneous. We have used these findings to introduce a model that generates instances with these key properties. In an average-case analysis, we have shown that both high clustering and heterogeneity of generated instances correlate with the effectiveness of the reduction rules.



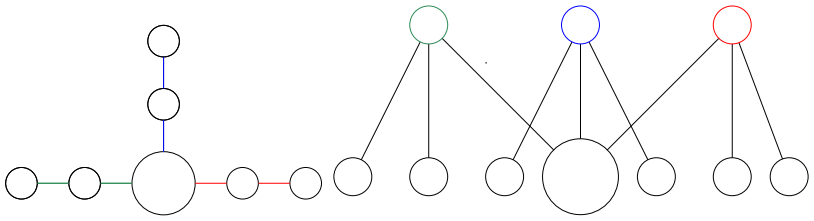
(a) Parameter $P_\triangle$

(b) Parameter C_4

(c) Parameter C_E

(d) Parameter C_S

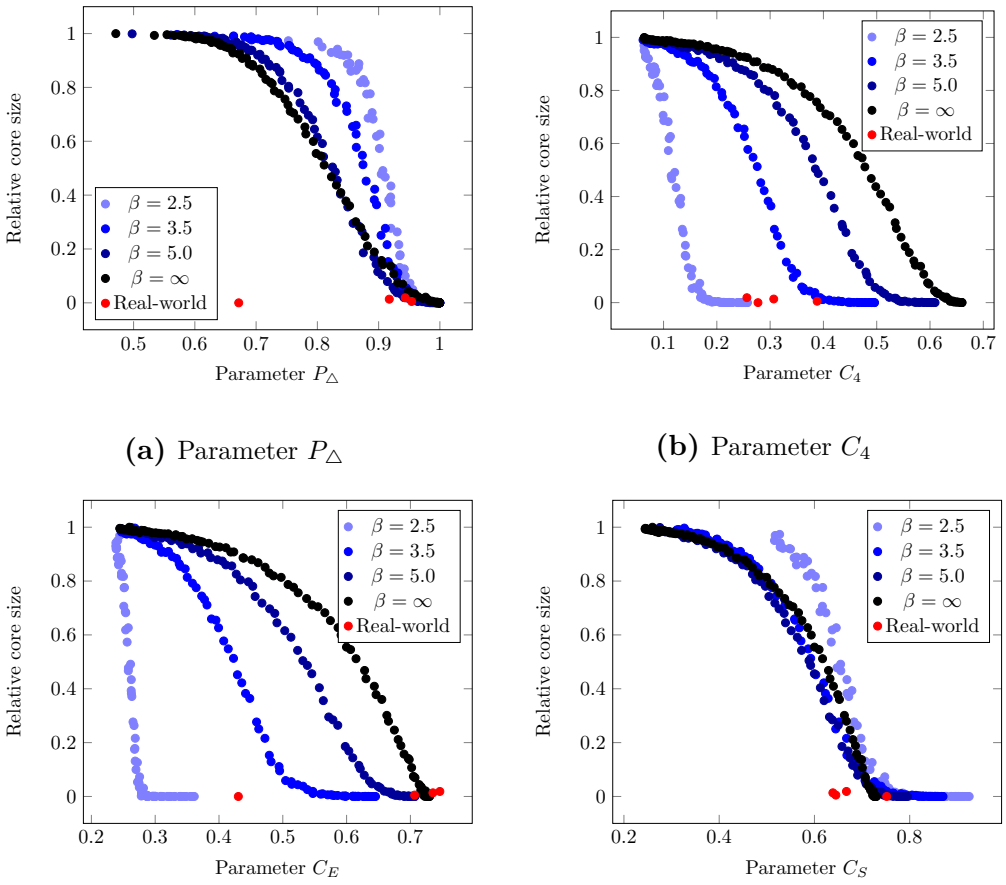
Figure 4.8: Plots for relation between the temperature T and the clustering parameters. All parameters are correlated to the temperature, but also show a dependence on the heterogeneity.



(a) A central station has many lines going through it.

(b) The bipartite representation of this structure contains paths of length 3, but no cycles.

Figure 4.9: A common structure found in real-world transit networks.



(a) Parameter $P_\triangle$

(b) Parameter C_4

(c) Parameter C_E

(d) Parameter C_S

Figure 4.10: Plots for the relation between the relative core size and the clustering parameters. There is a strong correlation, and the measured clustering of the real-world instances roughly matches the effectiveness predicted by our model.

5 Relation to Independent Set

In a graph $G = (V, E)$, an *independent set* is a subset of the vertices $S \subseteq V$ such that no two vertices in S are neighbors in G. Given a graph, the Maximum Independent Set problem asks for an independent set of maximum cardinality. Recall that a possible projection of a Hitting Set instance to a graph is the intersection graph of sets, that is, a graph with a vertex for each set and an edge between two set-vertices if and only if the two sets share an element. For a Hitting Set instance, we call a subsystem of sets that forms an independent set in the intersection graph an *independent set of sets*.

We describe a strong relation between the reduction rules for Hitting Set and the Maximum Independent Set problem on the intersection graph of sets. In particular, we show that the Hitting Set reduction rules are valid reduction rules for Maximum Independent Set on the intersection graph, too. Furthermore, we prove that for a Hitting Set instance that can be solved by reduction, the size of the maximum independent set of sets and the size of the minimum hitting set are equal. For our real-world instances, the gap between the two solution sizes is very small. This is shown in Table 5.1. We calculated the solution sizes by first applying the reduction rules and then using simple branching rules, namely, branching at the vertex/set that has the most neighbors.

Dataset	Independent set size	Hitting set size	Difference
NL	964	977	13
VBB	486	493	7
SNCF	275	279	4
DB	20	20	0

Table 5.1: Overview of the datasets and the solution sizes of the Maximum Independent Set and Hitting Set problem

This relation shows that, in order to better understand Path Cover by Vertices, the influence of the reduction rules on Maximum Independent Set should be considered. We only sketch the connection here and leave a thorough treatment for future work.

Lemma 5.1. *In a set system, the size of every independent set of sets is less than or equal to the size of every hitting set.*

Proof. For each set represented in the independent set of sets, at least one element has to be in the hitting set. None of the sets share elements, thus any hitting set must contain at least one element for each set in the independent set of sets. □

Thus, the size of any independent set is a lower bound on the minimum hitting set size, and the size of any hitting set is an upper bound on the maximum independent set of sets. Not only is there a connection between the solution sizes of both problems, but also a connection between the reduction rules of Hitting Set and their effect on independent sets of sets, as the following Lemma shows.

Lemma 5.2. *Let H be a family of sets and H' the family of sets after applying the* Hitting Set *reduction rules on H. Then, any independent set of sets in H' is also an independent set of sets in H.*

Proof. We prove this by contradiction. Assume the two sets s_1 and s_2 have no common element in H', but had elements in common in H. Then, all of these common elements must have been removed during the reduction. Let e_1 be the last element removed. Necessarily, e_1 was removed through an element reduction, so there was an element e_2 with $S(e_1) \subseteq S(e_2)$. Thus, s_1 and s_2 also had the element e_2 in common, which is a contradiction to our assumption that e_1 is the last common element that is deleted. □

If a Hitting Set instance can be solved by reduction, only sets with one element each remain. These form an independent set of sets in the core. By Lemma 5.2, they were an independent set of sets in the initial instance too. The corresponding elements form a minimum hitting set. By Lemma 5.1, the size of this hitting set is an upper bound on the maximum independent set of sets. This leads us to the following corollary.

Corollary 5.3. *For a* Hitting Set *instance can be solved by reduction, the size of the minimum hitting set and the size of the maximum independent set of sets is equal.*

Interestingly, this also means that if a Hitting Set instance can be solved by reduction, it also returns a certificate that this is the best possible solution in the form of an independent set of sets of the same size. Another consequence is the finding that in order for the Hitting Set instance to be solvable by reduction, the corresponding Maximum Independent Set instance on the intersection graph of sets has to be easy to solve too, in particular by the same reduction rules. To better understand this relation, we examine the conditions and effects of the reduction rules on the intersection graph of sets.

First, we look at element reduction. Recall that an element e_2 dominates e_1 if $S(e_1) \subseteq S(e_2)$. In the intersection graph of the sets, each element e creates the clique $S(e)$. Thus, the vertices $S(e_2)$ form a clique, so removing the element e_1 from all sets in the Hitting Set instance does not change the intersection graph.

Now we examine set reduction. Recall that a set s_1 dominates s_2 if $s_1 \subseteq s_2$. Since $s_1 \neq \emptyset$, the two sets share at least one element and thus are neighbors in the intersection graph. The neighbors of s_1 are the sets $\{s \mid s_1 \cap s \neq \emptyset\}$. Since $s_1 \subseteq s_2$, this means that s_2 shares all of these neighbors. Removing the dominated set s_2 is a valid reduction rule with respect to the Maximum Independent Set problem. Since s_1 and s_2 are neighbors, at most one of them can be part of the independent set of sets. For every independent set S that contains s_2, the set $(S \setminus \{s_2\}) \cup \{s_1\}$ is a valid independent set of sets too: s_1 now only blocks sets that were blocked by s_2 before. Therefore, we do not change the size of the maximum independent set of sets by removing s_2.

We have established that for a Hitting Set instance that can be solved by reduction, the size of the maximum independent set of sets and the size of the minimum hitting set are identical. We have also shown that in order to better understand the Path Cover by Vertices and Hitting Set problem, we should understand the Maximum Independent Set problem and the reduction rules. In addition, we note that for the real-world instances examined, the two sizes are in fact surprisingly close. Since the difference is very small for real-world instances, a parameterized algorithm for Hitting Set in this difference would seem helpful. Sadly, such an FPT algorithm is rather improbable by the following argument. The difference is always less than or equal to the size of the minimum hitting set. But since there is no FPT algorithm for Hitting Set in the solution size unless W[2]=FPT, the same condition holds for an FPT algorithm for Hitting Set in this difference. Nonetheless, this finding indicates that a better understanding of the Maximum Independent Set problem and its reduction rules might

give a better understanding of the Hitting Set problem and eventually lead to algorithms that utilize these insights.

6 Conclusion

In this work we researched the problem Path Cover by Vertices, which, given a set of paths in a graph, asks for a minimum number of vertices such that each path is covered. This was motivated by the findings of Weihe [38], who showed that this problem can be solved almost completely for real-world transit data by applying simple reduction rules. We showed that instances on forests and cycle graphs can be solved in polynomial time. While this suggested a parameterization by tree-likeness, we showed that there is no FPT algorithm for Path Cover by Vertices in the treewidth or feedback vertex number unless P=NP. We showed that Path Cover by Vertices is easy to solve for graphs of treewidth 1 and feedback vertex number 0 (forests), while instances on graphs of treewidth 3 or feedback vertex number 2 are NP-hard. It remains an open problem what the runtime for Path Cover by Vertices on graphs of treewidth 2 and graphs of feedback vertex number 1 is. Based on these results, we changed our focus to the Hitting Set problem, which is derived by treating each path as a set of elements. Again, we showed that this view of the problem is NP-hard even for very restricted instance classes. Additionally, we have shown that finding a graph representation for a Hitting Set instance that uses a minimum number of edges is NP-hard.

This indicates that a parameterized complexity analysis in the typical worst-case fashion fails both for Path Cover by Vertices and Hitting Set. In an empirical average-case analysis, we proposed and showed that real-world instances of transit data show strong clustering and heterogeneous element degrees. We introduced a model for generating Hitting Set instances that exhibit both of these structural properties in a configurable way. In our model, we assign each element and set a weight as well as a position on a circle. An element is part of a set based on a probability that uses the weights and the distance of the positions of the element and set. By assigning the element weights according to a power-law distribution and by utilizing a temperature

in the probability function, we can control the heterogeneity and the clustering of the generated graph.

We introduced the relative core size metric and used it to examine the influence of both clustering and heterogeneity on the effectiveness of the reduction rules. The analysis indicated that both the heterogeneity and the clustering are main causes of the effectiveness of the rules. While the effect of the heterogeneity decreases for instances with small clustering, the positive effect of clustering on the core size still holds even for homogeneous instances. We compared several clustering metrics and their behavior in different parameter settings and showed that the effectiveness observed on the real-world data is very close to what we would expect based on the experiments with generated instances. The empirical findings give us a better understanding of the sources of effectiveness of the reduction rules on real-world instances and lay the foundations for future work. It might be insightful to examine a more realistic model that utilizes two-dimensional geometry as a basis. Another possibility is to use a different measure of the core size, e.g., by measuring the size of the largest remaining component. We hypothesize that similar findings on the clustering of Hitting Set instances in other fields, such as computational biology and Boolean logic, are probable. Additionally, a theoretical average-case analysis on the effect of the clustering for our specific model could yield valuable insights. A theoretical finding in terms of a parameterization by clustering is unlikely, since the clustering coefficients can be changed almost arbitrarily by adding structures that have a very high or low clustering based on the used metric. However, an analysis of a simpler model seems more feasible.

Lastly, we presented an interesting relation between the reduction rules and the Maximum Independent Set problem. The reduction rules are valid reduction rules for the Maximum Independent Set problem on the intersection graph of the sets, and each independent set of sets forms a lower bound on the Hitting Set solution size. We showed that the two values are very close for real-world instances. One consequence of this is that a better understanding of the Maximum Independent Set problem and its reduction rules could also lead to a better understanding of the Hitting Set problem. Similar to our approach to the Hitting Set problem, this could be approached by considering which instances of Maximum Independent Set are easy to solve and whether these findings can be generalized to a parameterized algorithm. Furthermore, a similar model for realistic Maximum Independent Set instances could be introduced in order to study the effect of clustering on the effectiveness of the reduction rules.

References

[1] F. N. Abu-Khzam. A kernelization algorithm for d-hitting set. *Journal of Computer and System Sciences*, 76(7):524–531, 2010. (Cited on page 16.)

[2] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47, 2002. (Cited on page 21.)

[3] J. Alstott, E. Bullmore, and D. Plenz. powerlaw: a python package for analysis of heavy-tailed distributions. *PLOS One*, 9(1):e85777, 2014. (Cited on page 24.)

[4] G. Ausiello, A. D'Atri, and M. Protasi. Structure preserving reductions among convex optimization problems. *Journal of Computer and System Sciences*, 21(1):136–153, 1980. (Cited on page 17.)

[5] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999. (Cited on page 3.)

[6] A. Bogdanov, L. Trevisan, et al. Average-case complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1):1–106, 2006. (Cited on page 2.)

[7] K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976. (Cited on page 17.)

[8] K. Bringmann, R. Keusch, and J. Lengler. Geometric inhomogeneous random graphs. *arXiv preprint arXiv:1511.00576*, 2015. (Cited on pages 3 and 26.)

[9] K. Bringmann, L. Kozma, S. Moran, and N. Narayanaswamy. Hitting set for hypergraphs of low VC-dimension. In *Proceedings of the 24th Annual European Symposium on Algorithms (ESA)*, pages 1–18, 2016. (Cited on page 17.)

[10] J. Chen, I. A. Kanj, and G. Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 411(40-42):3736–3756, 2010. (Cited on page 16.)

[11] F. Chung and L. Lu. Connected components in random graphs with given expected degree sequences. *Annals of Combinatorics*, 6(2):125–145, 2002. (Cited on page 3.)

[12] A. Clauset, C. R. Shalizi, and M. E. Newman. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703, 2009. (Cited on page 2.)

[13] B. Courcelle. The monadic second-order logic of graphs. I. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. (Cited on page 15.)

[14] M. Dom. Algorithimic aspects of the consecutive-ones property. *Bulletin of the EATCS*, 98:27–59, 2009. (Cited on page 17.)

[15] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer Science & Business Media, 2012. (Cited on pages 2, 16, and 18.)

[16] P. Erdős and A. Rényi. On the evolution of random graphs. *Publication of the Mathematical Institute of the Hungarian Academy of Science*, 5(1): 17–60, 1960. (Cited on page 3.)

[17] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *Computer Communication Review*, volume 29, pages 251–262, 1999. (Cited on page 3.)

[18] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer Science & Business Media, 2006. (Cited on page 15.)

[19] X. Gabaix. Zipf's law for cities: an explanation. *The Quarterly Journal of Economics*, 114(3):739–767, 1999. (Cited on page 25.)

[20] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976. (Cited on page 16.)

[21] J. Guo and R. Niedermeier. Exact algorithms and applications for tree-like weighted set cover. *Journal of Discrete Algorithms*, 4(4):608–622, 2006. (Cited on page 17.)

[22] P. W. Holland and S. Leinhardt. Transitivity in structural models of small groups. *Comparative Group Studies*, 2(2):107–124, 1971. (Cited on page 3.)

[23] W.-L. Hsu and R. M. McConnell. Pc trees and circular-ones arrangements. *Theoretical Computer Science*, 296(1):99–116, 2003. (Cited on page 17.)

[24] M. O. Jackson. *Social and economic networks*. Princeton University Press, 2010. (Cited on page 2.)

[25] B. M. Jansen. On structural parameterizations of hitting set: hitting paths in graphs using 2-SAT. *Journal of Graph Algorithms and Applications (JGAA)*, 21(2):219–243, 2017. (Cited on pages 14 and 18.)

[26] R. Karp. The probabilistic analysis of combinatorial optimization algorithms. In *Proceedings of the 10th International Symposium on Mathematical Programming (ISMP)*, 1979. (Cited on pages 2 and 20.)

[27] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972. (Cited on page 16.)

[28] D. Krioukov, F. Papadopoulos, M. Kitsak, A. Vahdat, and M. Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3): 036106, 2010. (Cited on page 3.)

[29] M. Latapy, C. Magnien, and N. Del Vecchio. Basic notions for the analysis of large two-mode networks. *Social Networks*, 30(1):31–48, 2008. (Cited on page 3.)

[30] G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*. Wiley interscience series in discrete mathematics and optimization. Wiley, 1988. (Cited on page 16.)

[31] R. Niedermeier and P. Rossmanith. An efficient fixed-parameter algorithm for 3-hitting set. *Journal of Discrete Algorithms*, 1(1):89–102, 2003. (Cited on page 16.)

[32] M. Penrose. *Random geometric graphs*. Number 5. Oxford university press, 2003. (Cited on page 3.)

[33] G. Robins and M. Alexander. Small worlds among interlocking directors: Network structure and distance in bipartite graphs. *Computational & Mathematical Organization Theory*, 10(1):69–94, 2004. (Cited on page 22.)

[34] N. Ruf and A. Schöbel. Set covering with almost consecutive ones property. *Discrete Optimization*, 1(2):215–228, 2004. (Cited on page 17.)

[35] A. Schöbel. Set covering problems with consecutive ones property. Technical report, Universität Kaiserslautern, 2001. (Cited on page 17.)

[36] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994. (Cited on page 21.)

[37] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440, 1998. (Cited on pages 3 and 21.)

[38] K. Weihe. Covering trains by stations or the power of data reduction. *Proceedings of the Algorithms and Experiments Conference (ALEX)*, pages 1–8, 1998. (Cited on pages 1, 8, 9, 18, 23, and 38.)

Declaration of Authorship

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references.

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst habe. Alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken wurden als solche kenntlich gemacht.

Potsdam, January 31, 2018

Philipp Fischbeck