The Minimum Feasible Tileset problem

Yann Disser*, Stefan Kratsch**, and Manuel Sorge***

Technische Universität Berlin, Germany.
{yann.disser|stefan.kratsch|manuel.sorge}@tu-berlin.de

Abstract. We consider the Minimum Feasible Tileset problem: Given a set of symbols and subsets of these symbols (scenarios), find a smallest possible number of pairs of symbols (tiles) such that each scenario can be formed by selecting at most one symbol from each tile. We show that this problem is NP-complete even if each scenario contains at most three symbols. Our main result is a 4/3-approximation algorithm for the general case. In addition, we show that the Minimum Feasible Tileset problem is fixed-parameter tractable both when parameterized with the number of scenarios and with the number of symbols.

1 Introduction

Consider the general assignment problem where several devices (e.g., workers, robots, microchips, ...) each can be used in one of k functions/modes (e.g., employing different skills, tools, instruction sets, ...) at a time. Given a set of scenarios, the goal is to assign k different functions to each device, such that, for each scenario, all functions requested by the scenario are available simultaneously. In this paper, we initiate the study of this problem for $k = 2$ and the case that each function is requested at most once by each scenario. Formally, we study the following problem (we use "*tile*" instead of "*device*" to intuitively capture the fact that a device/tile has two modes/sides).

> Minimum Feasible Tileset
> **Input:** A universe of symbols F, scenarios $\mathcal{S} \subseteq 2^F \setminus \{F\}$, and $\ell \in \mathbb{N}$.
> **Problem:** Is there a tileset $\mathcal{T}$ of at most ℓ tiles $T \in \binom{F}{2}$ that is feasible for all scenarios in $\mathcal{S}$?

In the above, we refer to (multi-)sets of tiles as *tilesets*. A tileset $\mathcal{T}$ is *feasible* for scenario S, if we can produce all symbols in S by taking at most one symbol from each tile in $\mathcal{T}$. Formally, a tileset $\mathcal{T}$ is feasible for a scenario $S \subset F$ if there is a mapping $\phi\colon \mathcal{T} \to F$, such that $\phi(T) \in T$ for all $T \in \mathcal{T}$, and $S \subseteq \phi[\mathcal{T}] := \{\phi(T) \mid T \in \mathcal{T}\}$. By definition, no scenario contains all symbols of F. Note that such a scenario would require $|F|$ tiles, making the problem trivial. Similarly, we may assume that all symbols in F appear in at least one scenario, otherwise we

* Supported by the Alexander von Humboldt-Foundation.
** Supported by the German Research Foundation (DFG), KR 4286/1.
*** Supported by the German Research Foundation (DFG), NI 369/12.

can simply remove each symbol that does not occur in any scenario. Finally, the requirement that tiles contain no less than two symbols can be met by arbitrarily assigning a second symbol to all tiles of cardinality one.

Apart from practical motivations Minimum Feasible Tileset is appealing from a structural point of view. In this work we exhibit equivalent definitions for the problem which are interesting in their own right. At first glance, Minimum Feasible Tileset is a covering problem since we must cover all scenarios using tiles that can each cover one of their two symbols in each scenario. It turns out that the problem can also be phrased as a packing/partitioning problem, but with an objective function different from the classical one in terms of number of packed objects or sets (see Section 3). In addition, having tiles be symbol sets of size two suggests a graph interpretation where we are asked to find a minimum set of edges such that for each scenario there is an orientation where each vertex has indegree at least one. We favor the tileset formulation, since it most naturally generalizes to the original assignment problem with tiles of larger sizes and scenarios which contain multiple copies of the same symbols. Also, the Minimum Feasible Tileset interpretation appears suitable for studying the effect of parameters, such as the number of symbols/scenarios, on the complexity.

Results and Outline. We analyze the structure of the graph that has the tiles of a minimum cardinality tileset as its edges, and show that this graph is always (wlog.) a forest. In fact, only the component structure of this forest matters: We may replace trees by arbitrary trees spanning the same components without affecting the feasibility of the corresponding tileset (Section 2). This lets us view Minimum Feasible Tileset as a partitioning problem, which in turn allows us to prove NP-completeness even when scenarios have size at most three (Section 3). As our main result, we complement the hardness with a 4/3-approximation algorithm (for scenarios of arbitrary sizes) inspired by the component structure of the optimum solution (Section 4). Finally, we show that the problem is fixed-parameter tractable with respect to the number of scenarios (Section 5) and the number of symbols (Section 6), respectively. Due to space constraints, we defer proofs for results marked by $\star$ to a full version of the paper.

Related Work. The problem most closely related to Minimum Feasible Tileset is arguably Set Packing, as 3-Set Packing appears as a subproblem in our approximation algorithm and also as the source problem for our NP-hardness reduction. Set Packing has been extensively studied for both approximability and parameterized complexity (see, e.g., [1,5,19] and [6,17] for some recent results). The main difference between the two problems is that Set Packing is a maximization problem whereas Minimum Feasible Tileset seeks to minimize the size of a feasible tileset—a measure that is only indirectly related to the number of sets (scenarios). In particular, Set Packing becomes trivial for a bounded number of sets, whereas for Minimum Feasible Tileset we get a nontrivial polynomial-time algorithm via integer linear programming.

As mentioned above, the Minimum Feasible Tileset problem can equivalently be seen as designing an edge-minimal graph on the set of symbols such that, for each scenario, the edges (tiles) can be oriented in such a way that all symbols in the scenario have indegree at least one. The question whether a *given* graph admits an orientation with certain properties has been studied in various settings. For example, Biedl et al. [2] proposed an approximation algorithm for finding a balanced acyclic orientation. Another natural constraint on an orientation that has been studied is to prescribe degrees for each vertex [8,10,14].

More abstractly, we are looking for a graph on the set of symbols that fulfills a certain constraint for each scenario. The case where the subgraph induced by each scenario has to be connected is well-studied [3,4,9,13,15]. In particular, it is NP-hard to find the minimum number of edges needed [9] and to decide whether a planar solution [3,15] or a solution of treewidth at most three [13] exists.

2 Graph structure of tilesets

The tiles in a tileset $\mathcal{T}$ over a universe of symbols F can be viewed as the edges of the undirected (multi-) graph $G(\mathcal{T}) := (F, \mathcal{T})$. In this section, we establish that there always exist optimal tilesets with a simple graph structure. This is made formal in the following lemma which will be useful in later sections.

Lemma 1 **($\star$).** *Let F be a universe of symbols, $\mathcal{S}$ a family of scenarios over F, and $\mathcal{T}$ a tileset feasible for $\mathcal{S}$. There is a tileset $\mathcal{T}' \subseteq \binom{F}{2}$ feasible for $\mathcal{S}$ such that $|\mathcal{T}'| \leq |\mathcal{T}|$ and $G(\mathcal{T}')$ is a forest.*

Note that each connected component of $G(\mathcal{T}')$ has size at least two because each symbol occurs in at least one scenario and hence is incident with at least one edge. We show that only the partition of the symbols induced by the component structure of a tileset matters, but not the exact topology of each of the trees.

Theorem 1 **($\star$).** *Let $\mathcal{S}$ be a family of scenarios and $\mathcal{T}$ be a tileset over symbols F. If $G(\mathcal{T})$ is a forest, then $\mathcal{T}$ is feasible for $\mathcal{S}$ if and only if no connected component C of $G(\mathcal{T})$ is fully contained in any scenario $S \in \mathcal{S}$, i.e., $C \nsubseteq S$ for all scenarios $S \in \mathcal{S}$ and all connected components C of $G(\mathcal{T})$.*

3 NP-Completeness of Minimum Feasible Tileset

In this section we establish the following completeness result.

Theorem 2. Minimum Feasible Tileset *is* NP*-complete, even if each scenario has size at most three.*

Let us check that Minimum Feasible Tileset is contained in NP: A feasible tileset can be encoded using polynomially many bits with respect to $|F|$. Verifying feasibility comes down to solving one bipartite matching problem for each scenario on an auxiliary graph that has an edge between each symbol in the scenario and every tile containing that symbol, which is possible in polynomial time.

It remains to prove NP-hardness. For this, we first give a reduction from the following partition problem, and later prove this problem to be NP-hard.

FINE CONSTRAINED PARTITION
Input: A universe U, constraints $\mathcal{V} \subseteq 2^U \setminus U$, and $p \in \mathbb{N}$.
Problem: Is there a partition $\mathcal{P}$ of U, $|\mathcal{P}| \geq p$, such that $P \nsubseteq V$ for all parts $P \in \mathcal{P}$ and all $V \in \mathcal{V}$?

Lemma 2. MINIMUM FEASIBLE TILESET *and* FINE CONSTRAINED PARTITION *are equivalent if we identify scenarios and constraints.*

Proof. We claim that an instance $(F, \mathcal{S}, \ell)$ of MINIMUM FEASIBLE TILESET admits a solution if and only if the instance $(F, \mathcal{S}, |F| - \ell)$ of FINE CONSTRAINED PARTITION admits a solution.

"$\Rightarrow$": By Lemma 1 there is a feasible tileset $\mathcal{T}'$ for $\mathcal{S}$ of cardinality at most ℓ such that $G(\mathcal{T}')$ is a forest. The connected components $C_1, \dots, C_k$ of $G(\mathcal{T}')$ induce a partition that is a solution for the FINE CONSTRAINED PARTITION instance: By Theorem 1 we indeed have $C_i \nsubseteq S$ for all connected components C_i, $i \in [k]$, and scenarios $S \in \mathcal{S}$. Furthermore, since there are at most ℓ edges in $G(\mathcal{T}')$ and each connected component is a tree, we have $\ell \geq \sum_{i=1}^{k} |C_i| - 1 = |F| - k$. Hence, our partition has at least $k \geq |F| - \ell$ parts.

"$\Leftarrow$": Let $\mathcal{P} = \{P_1, \dots, P_p\}$ be a solution for the FINE CONSTRAINED PARTITION instance. We construct a tileset $\mathcal{T}$ by setting $G(\mathcal{T})[P_i]$ to an arbitrary spanning tree for each $i \in [p]$. Since $P_i \nsubseteq S$ for each $S \in \mathcal{S}$ and each $i \in [p]$, by Theorem 1, $\mathcal{T}$ is feasible for $\mathcal{S}$. The number of tiles in $\mathcal{T}$ is $\sum_{i=1}^{p} |P_i| - 1 = |F| - p \leq |F| - (|F| - \ell) = \ell$, as required. □

Note that the corresponding optimization problems are dual to each other in the sense that one is to minimize ℓ and the other to maximize $|F| - \ell$. We are now ready to give a reduction to FINE CONSTRAINED PARTITION from EXACT COVER BY 3-SETS, which is well known to be NP-hard [12], hence, completing the proof of Theorem 2

EXACT COVER BY 3-SETS
Input: A universe X and a family $\mathcal{C}$ of three-element sets $C \in \binom{X}{3}$.
Problem: Is there an *exact cover* for X, i.e., a partition of X into a family $\mathcal{C}' \subseteq \mathcal{C}$ of disjoint sets?

Lemma 3. *There is a polynomial-time reduction from* EXACT COVER BY 3-SETS *to* FINE CONSTRAINED PARTITION *with constraints of size at most three.*

Proof. Let an instance $(X, \mathcal{C})$ of EXACT COVER BY 3-SETS be given. Without loss of generality, we may assume that $|X| = 3q$ for some integer q, as otherwise no exact cover exists. We construct an instance of FINE CONSTRAINED PARTITION with universe X asking for a partition of size at least q. First, we add constraints $\mathcal{V}_2 = \binom{X}{2}$ that exclude every two-element subset of X from all solution partitions. Since every solution partition needs to contain at least q parts and $|X| = 3q$, each such partition consists of sets of size exactly three. Next, we exclude partitions that contain sets outside of $\mathcal{C}$ by simply adding the constraints $\mathcal{V}_{\bar{\mathcal{C}}} = \binom{X}{3} \setminus \mathcal{C}$. This concludes the construction of the FINE CONSTRAINED PARTITION instance $(X, \mathcal{V}_2 \cup \mathcal{V}_{\bar{\mathcal{C}}}, q)$. Clearly, this takes polynomial time.

Now, if there is a partition $\mathcal{P}$ with at least q parts for the Fine Constrained Partition instance, by the above, we know that each of its parts is a set in $\mathcal{C}$. Hence $\mathcal{P}$ is an exact cover of X for family $\mathcal{C}$. Conversely, let $\mathcal{C}' \subseteq \mathcal{C}$ be an exact cover for X. Then $|\mathcal{C}'| \geq q$ and for all $C \in \mathcal{C}'$ and $V \in \mathcal{V}_2 \cup \mathcal{V}_{\bar{\mathcal{C}}}$ we have $C \nsubseteq V$, because C has size three and is not from $\mathcal{V}_{\bar{\mathcal{C}}}$. Hence also the Fine Constrained Partition instance has a solution. ◻

4 A 4/3-approximation for Minimum Feasible Tileset

In this section, we propose an approximation algorithm for Minimum Feasible Tileset with unbounded scenario size. Motivated by the structural insights of Section 2, we construct a tileset that induces a forest in the corresponding graph, with the property that none of its components are contained in a single scenario. Since a component of size k requires $k-1$ tiles, we additionally aim for small components in order to keep the resulting tileset small.

We first take as many components of size two as possible among all disjoint sets of two symbols that are not both contained in the same scenario. This can easily be achieved by computing a maximum matching in the graph that has an edge for each candidate component. Similarly, among all remaining symbols, we try to form many (disjoint) components of size three, without creating components that are contained in a single scenario. For this, we employ a simple greedy strategy, that repeatedly takes any possible component until no possible candidates remain. (While there are better packing strategies available for sets of size three, we will see that improving the packing strategy alone does not improve our approximation ratio.) Finally, for each leftover symbol we add an individual tile (pairing that symbol in such a way as to prevent cycles).

We give a more formal listing in Algorithm A. We use $\bar{F}_i(F') = \{C \in \binom{F'}{i} \mid \forall S \in \mathcal{S}\colon C \nsubseteq S\}$ to denote the family of all sets of symbols in F' that are of size i and not fully contained in a single scenario. In the following, we identify connected components with their sets of vertices.

Algorithm A**:** 4/3-approximation for minimum feasible tilesets

---

Input: A set F of symbols and a set $\mathcal{S}$ of scenarios, where $\mathcal{S} \subseteq 2^F \setminus \{F\}$.
Output: A set of tiles $\mathcal{T}$.
$\mathcal{T}_2 \leftarrow$ maximum matching in graph $G(\bar{F}_2(F))$.
$\mathcal{P} \leftarrow$ greedy set packing of $\bar{F}_3(F \setminus \bigcup_{t \in \mathcal{T}_2} t)$.
$\mathcal{T}_3 \leftarrow \bigcup_{\{f_1,f_2,f_3\} \in \mathcal{P}} \{\{f_1,f_2\},\{f_2,f_3\}\}$.
if $\mathcal{T}_2 \cup \mathcal{T}_3 \neq \emptyset$ **then** take $f_{\text{root}} \in \bigcup_{t \in \mathcal{T}_2 \cup \mathcal{T}_3} t$ **else** take $f_{\text{root}} \in F$.
$\mathcal{T}_1 \leftarrow \{\{f, f_{\text{root}}\} \mid f \in F \setminus \bigcup_{t \in \mathcal{T}_2 \cup \mathcal{T}_3} t\ , f \neq f_{\text{root}}\}$.
return $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T}_3$.

Theorem 3. *Algorithm A computes a 4/3-approximation for* Minimum Feasible Tileset*.*

Proof. We first argue that the set of tiles $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T}_3$ computed by Algorithm A is feasible for $\mathcal{S}$. First observe that $G(\mathcal{T})$ is a forest. This is true, because $G(\mathcal{T}_2 \cup \mathcal{T}_3)$ consists of trees of sizes 2 and 3, $G(\mathcal{T}_1)$ is a star, and $\mathcal{T}_1 \cap (\mathcal{T}_2 \cup \mathcal{T}_3)$ contains at most one node (f_{root}). Using Theorem 1 it only remains to show that no connected component C of $G(\mathcal{T})$ is contained in any scenario $S \in \mathcal{S}$, i.e. $C \cap S \subsetneq C$. By definition of Algorithm A this is true for all connected components of the graph $G(\mathcal{T}_2 \cup \mathcal{T}_3)$. If $\mathcal{T}_2 \cup \mathcal{T}_3 \neq \emptyset$, then each component of $G(\mathcal{T})$ is a superset of a component of $G(\mathcal{T}_2 \cup \mathcal{T}_3)$, and is thus not contained in any scenario. If $\mathcal{T}_2 \cup \mathcal{T}_3$ is empty, then $G(\mathcal{T}) = G(\mathcal{T}_1)$ consists of a single component that is not contained in any scenario, since, by definition, $F \notin \mathcal{S}$. Thus $\mathcal{T}$ is feasible for $\mathcal{S}$.

We now bound the size of $\mathcal{T}$ with respect to a minimum cardinality tileset $\mathcal{T}^\star$. To do this we *distribute* virtual currency *(gold)* to the symbols in F, such that the total gold distributed is 4/3 times the size of $\mathcal{T}^\star$. We later use this gold to *pay* one unit of gold to certain symbols that these can in turn use to *provide for* (at most) one tile of $\mathcal{T}$ that involves this symbol. To complete the proof, we establish that each tile of $\mathcal{T}$ is provided for by one of its two symbols.

Let $G^\star := G(\mathcal{T}^\star)$ be the graph induced by $\mathcal{T}^\star$ and $\bar{F}_i^\star$ be the set of connected components of size $i \in \{2, \ldots, |F|\}$ in $G^\star$. By Lemma 1, we may assume that $G^\star$ is a forest. Furthermore, because each symbol appears in at least one scenario, graph $G^\star$ does not contain components of size 1. Since the symbols in a component of size $i > 1$ are part of exactly $i - 1$ tiles in $\mathcal{T}^\star$, we may distribute all available gold by giving $4/3 \cdot \frac{i-1}{i}$ gold to each symbol in a component of $\bar{F}_i^\star$, for all $i \in \{2, \ldots, |F|\}$. This gold is used to pay symbols in what follows. We call a symbol $s \in F$ *sufficiently paid* if one of the following holds: (i) s is paid, (ii) s appears in a tile $T \in \mathcal{T}_2$ and the other symbol of T is paid, or (iii) s appears in a tile $T \in \mathcal{T}_3$ and the other two symbols in the same component of $G(\mathcal{T}_3)$ are paid. Below, we show how to sufficiently pay all symbols. This completes the proof, since then all tiles in $\mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T}_3$ can be provided for (note that then each tile in $\mathcal{T}_1$ contains its own paid symbol). We call a component of $G^\star$ *sufficiently paid*, if all its symbols are sufficiently paid. Let $F_{\geq 4}^\star := F \setminus \bigcup_{C \in \bar{F}_2^\star \cup \bar{F}_3^\star} C$ be the set of all symbols not in components of size two or three in $G^\star$. In paying the symbols we will maintain the invariant that each element of $\bar{F}_2^\star \cup \bar{F}_3^\star \cup F_{\geq 4}^\star$ is either sufficiently paid, or it still holds its gold (all its symbols still hold their gold, respectively).

We define a graph $H = (V, E)$ that has the components in $\bar{F}_2^\star \cup \bar{F}_3^\star$ as its vertices, as well as the symbols that are not part of these components, i.e., $V = \bar{F}_2^\star \cup \bar{F}_3^\star \cup F_{\geq 4}^\star$. In this way, each vertex of H represents up to three symbols. For each tile $T \in \mathcal{T}_2$ we introduce an edge connecting the vertices of H representing the two symbols of T, possibly introducing self-loops. Since $\mathcal{T}_2$ is a matching, and since the vertices in H represent at most three symbols each, all vertices in H have degree at most 3. We partition the edges of H into paths, cycles, and self-loops, and show for each how to use the gold remaining at its vertices to pay all symbols in the components of $G^\star$ that are intersected by the path/cycle/self-loop. We will ensure that every symbol (except possibly f_{root})

on a tile in $\mathcal{T}_1$ is paid. Since each symbol on a tile of $\mathcal{T}_2$ appears only exactly on this and no other tile of $\mathcal{T}_2 \cup \mathcal{T}_3$, it is thus sufficient to pay only one of the two symbols on each tile of $\mathcal{T}_2$.

Let $\mathcal{P}$ be the set of all paths in H connecting (different) vertices of degree 1 or 3 with internal nodes of degree 2. Consider the paths in $\mathcal{P}$ one by one. We use the gold available along path $P \in \mathcal{P}$ of length k as follows. Let N_2, N_3 be the number of internal nodes of P that represent 2 and 3 symbols, respectively. Note that P has no inner nodes that represent a single symbol, since $\mathcal{T}_2$ is a matching, and hence $k = 1 + N_2 + N_3$. Also, P is the only path visiting these inner nodes and hence they all still hold their gold. Let $N_1^{\text{end}}, N_2^{\text{end}}, N_3^{\text{end}} \leq 2$ be the number of endpoints of P that still hold gold and represent 1, 2, and 3 symbols, respectively. Similarly, let N_0^{end} be the number of endpoints without gold. By our invariant, the symbols or components represented by the endpoints without gold left have already been sufficiently paid before. We make sure that all other nodes along P are sufficiently paid. We do this by, for all tiles that form the path P, paying one of the two corresponding symbols, and, in addition, paying *every* further symbol represented by nodes along P. Note that this preserves the invariant. The total cost is

$$C^- = k + N_2^{\text{end}} + 2N_3^{\text{end}} + N_3 - N_0^{\text{end}} = 1 + N_2^{\text{end}} + 2N_3^{\text{end}} + N_2 + 2N_3 - N_0^{\text{end}}. \quad (1)$$

Using that each endpoint of P that contributes to N_1^{end} represents a symbol that is part of a component in $G^\star$ of size $i \geq 4$, we get that the gold available at this symbol is at least $\frac{4}{3} \cdot \frac{i-1}{i} \geq 1$. Hence, the gold available to us is at least

$$C^+ = \frac{4}{3}(N_2^{\text{end}} + 2N_3^{\text{end}} + N_2 + 2N_3 + \frac{3}{4}N_1^{\text{end}}). \quad (2)$$

Since $N_0^{\text{end}} + N_1^{\text{end}} + N_2^{\text{end}} + N_3^{\text{end}} = 2$, we get

$$C^+ - C^- = 1 - \frac{2}{3}N_2^{\text{end}} - \frac{1}{3}N_3^{\text{end}} + \frac{1}{3}N_2 + \frac{2}{3}N_3.$$

Hence, we have $C^+ \geq C^-$, unless $N_2^{\text{end}} = 2$ and $N_0^{\text{end}} = N_1^{\text{end}} = N_3^{\text{end}} = N_2 = N_3 = 0$, i.e. P is of length one, connecting two tiles $p_1, p_2 \in \bar{F}_2^\star$ by an edge which corresponds to a tile $t \in \mathcal{T}_2$. To see that this case cannot occur, observe that, first, p_1 and p_2 are of degree 1 in H. Second, since $\mathcal{T}^\star$ is feasible, no component of $G^\star$ is contained in a single scenario (Theorem 1), and thus $p_1, p_2 \in \bar{F}_2^\star \subseteq \bar{F}_2(F)$. This is a contradiction to $\mathcal{T}_2$ being a maximum matching in graph $G(\bar{F}_2(F))$, as the matching can be augmented by removing t and adding p_1 and p_2.

Similarly to the above, we can consider all cycles in H with at most one node of degree 3 one by one. (Note that cycles with at least two nodes of degree 3 contain a path as before.) If a cycle of length k does not contain a node of degree 3, or the node of degree 3 is not yet sufficiently paid (and thus still holds its gold), the cost for the cycle and its available gold are

$$C^- = k + N_3 = N_2 + 2N_3 = \frac{3}{4}C^+ < C^+,$$

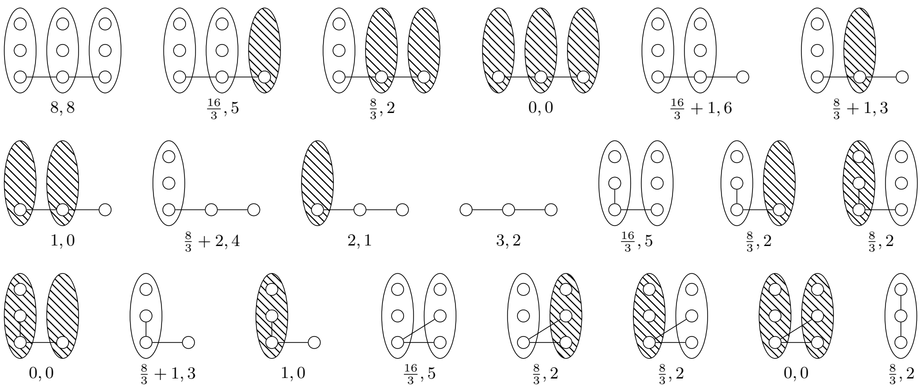

Fig. 1. Possible intersections of components of $G(\mathcal{T}_3)$ (arcs) and $G^\star$ (ellipses). Shaded components have been sufficiently paid previously. Configurations are labeled by the available gold C^+ and the required gold C^-. Symmetrical configurations are omitted.

where N_2, N_3 are the numbers of nodes of P that represent 2 and 3 symbols, respectively. If the node of degree 3 has no gold left, then it has already been sufficiently paid and $C^- = N_2 + 2N_3 - 3 < \frac{3}{4}C^+$. In either case, the available gold allows to sufficiently pay all nodes along the cycle. Finally, each self-loop in H connects two symbols in the same component C of size 2 or 3 in $G^\star$. If $|C| = 2$, the gold available among the two symbols is $C^+ = \frac{4}{3}$, while we require only $C^- = 1$ unit of gold. If $|C| = 3$, we have $C^+ = \frac{8}{3}$ and $C^- = 2$.

After processing all paths, cycles, and self-loops all nodes of H intersecting a tile of $\mathcal{T}_2$ are sufficiently paid. In particular, since $\mathcal{T}_2$ is a maximum matching, all components in $\bar{F}_2^\star$ are sufficiently paid. In the next step we ensure that all components of $\bar{F}_3^\star$ are sufficiently paid. By construction, every element of $\bar{F}_3^\star$, that is not sufficiently paid yet, intersects at least one tile of $\mathcal{T}_3$. We can thus consider the components of $G(\mathcal{T}_3)$ one by one and make sure to sufficiently pay each element of $\bar{F}_3^\star$ that intersects the considered component of $G(\mathcal{T}_3)$.

Consider a component of $G(\mathcal{T}_3)$ involving the three symbols f_1, f_2, f_3 (cf. Figure 1 in the following). Let $\mathcal{C}_3 \subseteq \bar{F}_3^\star$ be the set of components of size 3 in $G^\star$ that involve at least one of these symbols and have not yet been sufficiently paid (i.e., still hold their gold). Further, let N_n be the number of symbols among $\{f_1, f_2, f_3\} \cap F_{\geq 4}^\star$ that are not yet sufficiently paid. Since all components in $\bar{F}_2^\star$ are sufficiently paid, the gold we have available is at least $C^+ \geq \frac{4}{3}(2|\mathcal{C}_3| + \frac{3}{4}N_n)$. We ensure that (at least) two symbols among f_1, f_2, f_3 are paid, as well as all other symbols appearing in $\mathcal{C}_3$. In this way, each component in $\mathcal{C}_3$ is sufficiently paid. Note that this preserves our invariant that each element of $\bar{F}_2^\star \cup \bar{F}_3^\star \cup F_{\geq 4}^\star$ is either sufficiently paid, or still holds its gold. The cost for paying the symbols f_1, f_2, f_3 is at most 2. Since in addition to f_1, f_2, f_3 there are $3|\mathcal{C}_3| + N_n - 3$ symbols needing pay in $\bigcup_{C \in \mathcal{C}_3} C \cup \{f_1, f_2, f_3\}$, and because $|\mathcal{C}_3| \leq 3$, the total cost is

$$C^- \leq 3|\mathcal{C}_3| + N_n - 1 \leq \frac{8}{3}|\mathcal{C}_3| + N_n \leq C^+.$$

At this point, we have sufficiently paid all components in $\bar{F}_2^\star \cup \bar{F}_3^\star$ using gold only from these components. This means that all remaining symbols that are not sufficiently paid yet have at least $\frac{4}{3} \cdot \frac{4-1}{4} = 1$ gold available, which we can use to pay these symbols themselves. Now all elements of $\bar{F}_2^\star \cup \bar{F}_3^\star \cup F_{\geq 4}^\star$ have been sufficiently paid and the proof is complete. □

Our analysis of Algorithm A is tight in three different spots: (i) A path of length 1 in the graph H defined above that visits a component of size 2 and a component of size 3 of the optimum solution $\mathcal{T}$ may lead to 4 tiles in our solution compared to the 3 tiles required in the optimum solution, i.e., Equations (1) and (2) coincide if $N_2^{\text{end}} = N_3^{\text{end}} = 1$ and all other terms vanish. (ii) The first intersection of a component of $G(\mathcal{T}_3)$ with components of $G^\star$ illustrated in Figure 1 may lead to 8 tiles in our solution compared to the 6 tiles required in the optimum solution. (iii) Each symbol of a component of size 4 in $G^\star$ might result in a single tile for this symbol only, in which case the optimum solution requires 3 tiles for the symbols of the component, while our solution requires 4 tiles. To improve Algorithm A we have to address each of these three bottlenecks. For (i), we either would have to alter the matching $\mathcal{T}_2$ to prevent the described situation, or combine the analysis to account for the loss in other places. The aspect (ii) can easily be prevented by employing a more sophisticated set packing algorithm (e.g., the $(4/3+\varepsilon)$-approximation of Cygan [5]). Finally, to avoid (iii), we would need to pack sets of size 4 similarly to our packing of sets of size 3. In addition to requiring one more level of analysis, this would also complicate the other levels, as we would have to include sets of size 4 in our reasoning there.

5 Bounded number of scenarios

We prove that Minimum Feasible Tileset can be solved in polynomial time when the number $|\mathcal{S}|$ of scenarios is bounded. More precisely, we provide an algorithm that solves any instance $(F, \mathcal{S}, k)$ in time $f(|\mathcal{S}|)|(F, \mathcal{S}, k)|^c$, i.e., in time $\mathcal{O}(|(F, \mathcal{S}, k)|^c)$ for bounded values of $|\mathcal{S}|$. In other words, Minimum Feasible Tileset is fixed-parameter tractable with respect to the number of scenarios.

Our algorithm works by first translating the input instance $(F, \mathcal{S}, \ell)$ into an integer linear program (ILP) in such a way that the ILP is feasible (i.e., contains at least one integer point) if and only if $(F, \mathcal{S}, \ell)$ admits a feasible tileset with at most ℓ tiles. The ILP uses $\mathcal{O}(|\mathcal{S}|^{|\mathcal{S}|})$ variables. Lenstra [18] proved that deciding feasibility of any ILP is fixed-parameter tractable with respect to the number p of variables; the currently fastest algorithm has $\mathcal{O}^*(p^{\mathcal{O}(p)})$ running time and was obtained by Frank and Tardos [11], modifying an algorithm by Kannan [16]. Using this, we can prove the following result.

Theorem 4 ($\star$). Minimum Feasible Tileset *on instances with at most k scenarios can be solved in time $\mathcal{O}^*(k^{\mathcal{O}(k^{k+1})})$.*

6 Bounded number of symbols

We analyze the influence of the number of symbols $|F|$ on the complexity of solving an instance $(F, \mathcal{S}, \ell)$ of Minimum Feasible Tileset. It is easy to see that the problem becomes solvable in polynomial time when F is bounded: The instance is trivial if $\ell \geq |F|$ since, in that case, we can afford to dedicate a separate tile for each symbol. Otherwise, there are only $\mathcal{O}(|F|^{2\ell}) \subseteq \mathcal{O}(|F|^{2|F|})$ ways to fix ℓ tiles. As mentioned in Section 3, each candidate tileset can be verified by solving a bipartite matching problem for each scenario, on a graph that has an edge between each symbol in the scenario and every tile containing that symbol. This yields an overall runtime of $\mathcal{O}^*(|F|^{2|F|})$, and, hence, fixed-parameter tractability in $|F|$. Using structural insights of Section 2 we are able to improve on this naive running time.

Theorem 5 ($\star$). *Instances $(F, \mathcal{S}, \ell)$ of* Minimum Feasible Tileset *can be solved in time $\mathcal{O}^*(3^{|F|})$.*

Note that, as every symbol occurs in a scenario, $\ell \geq |F|/2$. Hence, Theorem 5 gives a fixed-parameter algorithm also for parameter ℓ.

After this fixed-parameter tractability result, and taking into account the trivial bound of $2^{|F|}$ for the number of scenarios (giving a worst-case size of instances of $\mathcal{O}(2^{|F|}|F|)$), it is natural to ask whether polynomial-time preprocessing can simplify input instances to size polynomial in $|F|$. We show that this is impossible unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$ (and the polynomial hierarchy collapses). More generally, we prove that for the restricted case d-Minimum Feasible Tileset, where scenarios have size at most d, no polynomial-time algorithm can achieve a size of $\mathcal{O}(k^{d-\varepsilon})$. Note that this restricted case has an essentially matching upper bound of $|\mathcal{S}| < (|F|+1)^d = \mathcal{O}(|F|^d)$.[1] As a consequence there is no reduction to size polynomial in $|F|$ for the general Minimum Feasible Tileset problem: Any size $\mathcal{O}(k^c)$ preprocessing for Minimum Feasible Tileset could be used for d-Minimum Feasible Tileset, for any $d > c$, and violate the lower bound.

Theorem 6. *Let $d \geq 3$ and ε be a positive real. There is no polynomial-time algorithm that reduces every instance of d-*Minimum Feasible Tileset *to an equivalent instance (possibly of a different problem) of size $\mathcal{O}(|F|^{d-\varepsilon})$, unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.*

To prove Theorem 6 we employ a similar result by Dell and Marx [6] for Exact Cover by d-Sets, which is defined as follows.[2]

Exact Cover by d-Sets
Input: A universe X and a family $\mathcal{C}$ of d-element sets $C \in \binom{X}{d}$.
Problem: Is there an *exact d-set cover* for X, i.e., a partition of X into a family $\mathcal{C}' \subseteq \mathcal{C}$ of disjoint sets?

[1] A compression to $\mathcal{O}(|F|^d)$ size can be achieved by specifying one bit for each possible scenario in $\mathcal{S}$ and setting it to one if the scenario is present and zero otherwise.

[2] Dell and Marx called this problem Perfect d-Set Matching.

Note that the original result by Dell and Marx [6] is given in terms of the size k of an exact d-set cover. Clearly, $k = \frac{|U|}{d}$ and, thus, we have $\mathcal{O}(k^{d-\varepsilon}) = \mathcal{O}(|U|^{d-\varepsilon})$ and may instead phrase the result in terms of $|U|$. Furthermore, their result builds on work by Dell and van Melkebeek [7] and, thus, extends to any polynomial time algorithms (rather than just kernels) whose output instances can be with respect to a different problem. We give the following paraphrased version of the result.

Theorem 7 (Dell and Marx [6]). *Let $d \geq 3$ and ε be a positive real. There is no polynomial-time algorithm that reduces every instance $(U, \mathcal{H})$ of* Exact Cover by d-Sets *to an equivalent instance of size $\mathcal{O}(|U|^{d-\varepsilon})$ (possibly with respect to a different problem), unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.*

The following lemma, together with Theorem 7, directly implies Theorem 6.

Lemma 4 ($\star$). *There is a polynomial-time reduction from* Exact Cover by d-Sets *to* Minimum Feasible Tileset *such that instances $(X, \mathcal{C})$ are mapped to instances $(F, \mathcal{S}, \ell)$ with $F = X$ and scenario size at most d.*

We now consider a more general setting: In the Generalized Minimum Feasible Tileset problem we are also given a set of symbols and a set of scenarios, but here each scenario may be a *multi-set* of symbols (or, equivalently, each scenario is a function $S\colon F \to \mathbb{N}$ indicating the number of copies of each symbol f needed for S). We prove that Generalized Minimum Feasible Tileset can be solved in time $\mathcal{O}^*(|F|^{\mathcal{O}(|F|^2)})$. Note that for this problem the solution size ℓ may be much larger than $|F|$ and similarly the number of scenarios cannot in general be bounded in $|F|$.

Theorem 8 ($\star$). Generalized Minimum Feasible Tileset *can be solved in time $\mathcal{O}^*(|F|^{\mathcal{O}(|F|^2)})$, i.e., it is fixed-parameter tractable with respect to $|F|$.*

7 Conclusion

We initiated the study of the Minimum Feasible Tileset problem and exposed an interesting combinatorial structure. We proved the problem to be NP-complete even in the restricted case with scenarios of size at most three. On the positive side, we showed that the Minimum Feasible Tileset problem admits a 4/3-approximation algorithm and that it is fixed-parameter tractable with respect to the number of scenarios and number of symbols. The latter algorithm works also for the Generalized Minimum Feasible Tileset problem where each scenario can contain multiple copies of a symbol and we believe that it can be further generalized to work also for the original assignment problem where also tiles of larger (but constant) size are allowed. It would be interesting to see whether our other positive results transfer to this more general setting. We note that our approximation algorithm relies heavily on the structural observations from Section 2 which do not seem to generalize well. Our integer linear program for a fixed number of scenarios does not seem easily adaptable either.

References

1. N. Bansal, A. Caprara, and M. Sviridenko. A new approximation method for set covering problems, with applications to multidimensional bin packing. *SIAM Journal on Computing*, 39(4):1256–1278, 2009.
2. T. Biedl, T. Chan, Y. Ganjali, M. Hajiaghayi, and D. Wood. Balanced vertex-orderings of graphs. *Discrete Applied Mathematics*, 148(1):27–48, 2005.
3. K. Buchin, M. J. van Kreveld, H. Meijer, B. Speckmann, and K. Verbeek. On planar supports for hypergraphs. *Journal of Graph Algorithms and Applications*, 15(4):533–549, 2011.
4. J. Chen, C. Komusiewicz, R. Niedermeier, M. Sorge, O. Suchý, and M. Weller. Effective and efficient data reduction for the subset interconnection design problem. In *Proceedings of the 24th International Symposium on Algorithms and Computation (ISAAC)*, pages 361–371, 2013.
5. M. Cygan. Improved approximation for 3-dimensional matching via bounded pathwidth local search. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 509–518, 2013.
6. H. Dell and D. Marx. Kernelization of packing problems. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 68–81, 2012.
7. H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 251–260, 2010.
8. Y. Disser and J. Matuschke. Degree-constrained orientations of embedded graphs. In *Proceedings of the 23rd International Symposium on Algorithms and Computation (ISAAC)*, pages 506–516, 2012.
9. D.-Z. Du and Z. Miller. Matroids and subset interconnection design. *SIAM Journal on Discrete Mathematics*, 1(4):416–424, 1988.
10. A. Frank and A. Gyárfás. How to orient the edges of a graph. *Colloquia mathematica societatis Janos Bolyai*, 18:353–364, 1976.
11. A. Frank and É. Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987.
12. M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, 1979.
13. G. Gottlob and G. Greco. On the complexity of combinatorial auctions: structured item graphs and hypertree decomposition. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC)*, pages 152–161, 2007.
14. S. Hakimi. On the degrees of the vertices of a directed graph. *Journal of the Franklin Institute*, 279(4):290–308, 1965.
15. D. S. Johnson and H. O. Pollak. Hypergraph planarity and the complexity of drawing Venn diagrams. *Journal of Graph Theory*, 11(3):309–325, 1987.
16. R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12:415–440, 1987.
17. I. Koutis. Faster algebraic algorithms for path and packing problems. In *Proceedings of the 35th International Colloquium on Automata (ICALP)*, pages 575–586, 2008.
18. H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983.
19. M. Sviridenko and J. Ward. Large neighborhood local search for the maximum set packing problem. In *40th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 792–803, 2013.