Parameterized Complexity of Stable Roommates with Ties and Incomplete Lists Through the Lens of Graph Parameters

Robert Bredereck
Technische Universität Berlin, Chair of Algorithmics and Computational Complexity, Germany
robert.bredereck@tu-berlin.de

Klaus Heeger
Technische Universität Berlin, Chair of Algorithmics and Computational Complexity, Germany
heeger@tu-berlin.de

Dušan Knop
Technische Universität Berlin, Chair of Algorithmics and Computational Complexity, Germany
Department of Theoretical Computer Science, Faculty of Information Technology,
Czech Technical University in Prague, Prague, Czech Republic
dusan.knop@fit.cvut.cz

Rolf Niedermeier
Technische Universität Berlin, Chair of Algorithmics and Computational Complexity, Germany
rolf.niedermeier@tu-berlin.de

Abstract

We continue and extend previous work on the parameterized complexity analysis of the NP-hard Stable Roommates with Ties and Incomplete Lists problem, thereby strengthening earlier results both on the side of parameterized hardness as well as on the side of fixed-parameter tractability. Other than for its famous sister problem Stable Marriage which focuses on a bipartite scenario, Stable Roommates with Incomplete Lists allows for arbitrary acceptability graphs whose edges specify the possible matchings of each two agents (agents are represented by graph vertices). Herein, incomplete lists and ties reflect the fact that in realistic application scenarios the agents cannot bring *all* other agents into a *linear* order. Among our main contributions is to show that it is W[1]-hard to compute a maximum-cardinality stable matching for acceptability graphs of bounded treedepth, bounded tree-cut width, and bounded feedback vertex number (these are each time the respective parameters). However, if we "only" ask for perfect stable matchings or the mere existence of a stable matching, then we obtain fixed-parameter tractability with respect to tree-cut width but not with respect to treedepth. On the positive side, we also provide fixed-parameter tractability results for the parameter feedback edge set number.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Graph algorithms analysis; Mathematics of computing → Matchings and factors

Keywords and phrases Stable matching, acceptability graph, fixed-parameter tractability, W[1]-hardness, treewidth, treedepth, tree-cut width, feedback set numbers

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2019.44

Related Version A full version of this paper can be found at `http://arxiv.org/abs/1911.09379`.

Funding Main work was done while all authors were with TU Berlin.
Klaus Heeger: Supported by DFG Research Training Group 2434 "Facets of Complexity".
Dušan Knop: Supported by the DFG, project MaMu (NI 369/19).

1 Introduction

The computation of stable matchings is a core topic in the intersection of algorithm design and theory, algorithmic game theory, and computational social choice. It has numerous applications – the research goes back to the 1960s. The classic (and most prominent from introductory textbooks) problem Stable Marriage, which is known to be solvable in linear time, relies on complete bipartite graphs for the modeling with the two sides representing the same number of "men" and "women". Herein, each side expresses preferences (linear orderings aka rankings) over the opposite sex. Informally, stability then means that no two matched agents have reason to break up. Stable Roommates, however, is not restricted to a bipartite setting: given is a set V of agents together with a preference list $\mathcal{P}_v$ for every agent $v \in V$, where a preference list $\mathcal{P}_v$ is a strict (linear) order on $V \setminus \{v\}$. The task is to find a *stable matching*, that is, a set of pairs of agents such that each agent is contained in at most one pair and there is no blocking edge (i.e., a pair of agents who strictly prefer their mates in this pair to their partners in the matching; naturally, we assume that agents prefer to be matched over being unmatched). Such a matching can be computed in polynomial time [18]. We refer the reader to the monographs [16, 25] for a general discussion on Stable Roommates. Recent practical applications of Stable Roommates and its variations also to be studied here range from kidney exchange to connections in peer-to-peer networks [10, 33, 34].

If the preference lists $\mathcal{P}_v$ for all agents v are complete, then the graph-theoretic model behind is trivial – a complete graph reflects that every agent ranks all other agents. In the more realistic scenario that an agent may only rank part of all other agents, the corresponding graph, referred to as *acceptability graph*, is no longer a complete graph but can have an arbitrary structure. We assume that the acceptability is symmetric, that is, if an agent v finds an agent u acceptable, then also agent u finds v acceptable. Moreover, to make the modeling of real-world scenarios more flexible and realistic, one also allows ties in the preference lists (rankings) of the agents, meaning that tied agents are considered equally good. Unfortunately, once allowing ties in the preferences, Stable Roommates already becomes NP-hard [24, 32], indeed this is true even if each agent finds at most three other agents acceptable [7]. Hence, in recent works specific (parameterized) complexity aspects of Stable Roommates with Ties and Incomplete Lists (SRTI) have been investigated [1, 3, 5]. In particular, while Bredereck et al. [3] studied restrictions on the structure of the preference lists, Adil et al. [1] initiated the study of structural restrictions of the underlying acceptability graph, including the parameter treewidth of the acceptability graph. We continue Adil et al.'s line of research by systematically studying three variants ("maximum", "perfect", "existence") and by extending significantly the range of graph parameters under study, thus gaining a fairly comprehensive picture of the parameterized complexity landscape of Stable Roommates with Ties and Incomplete Lists.

Notably, while previous work [1, 15] argued for the (also practical) relevance for studying the structural parameters treewidth and vertex cover number, our work extends this to further structural parameters that are either stronger than vertex cover number or yield more positive algorithmic results than possible for treewidth. We study the arguably most natural optimization version of Stable Roommates with ties and incomplete lists, referred to as Max-SRTI:

Max-SRTI

Input:	A set V of agents and a profile $\mathcal{P} = (\mathcal{P}_v)_{v \in V}$.
Task:	Find a maximum-cardinality stable matching or decide that none exists.

In addition to Max-SRTI, we also study two NP-hard variants. The input is the same, but the task either changes to finding a *perfect* stable matching – this is Perfect-SRTI – or to finding just *any* stable matching – this is SRTI-Existence.[1]

Perfect-SRTI

Input: A set of agents V and a profile $\mathcal{P} = (\mathcal{P}_v)_{v \in V}$.
Task: Find a perfect stable matching or decide that none exists.

SRTI-Existence

Input: A set of agents V and a profile $\mathcal{P} = (\mathcal{P}_v)_{v \in V}$.
Task: Find a stable matching or decide that none exists.

Related Work. On bipartite acceptability graphs, where Stable Roomates is called Stable Marriage, Max-SRTI admits a polynomial-time factor-$\frac{2}{3}$-approximation [29]. However, even on bipartite graphs it is NP-hard to approximate Max-SRTI by a factor of $\frac{29}{33}$, and Max-SRTI cannot be approximated by a factor of $\frac{3}{4} + \epsilon$ for any $\epsilon > 0$ unless Vertex Cover can be approximated by a factor strictly smaller than two [36]. Note that, as we will show in our work, SRTI-Existence is computationally hard in many cases, so good polynomial-time or even fixed-parameter approximation algorithms for Max-SRTI seem out of reach.

Perfect-SRTI was shown to be NP-hard even on bipartite graphs [19]. This holds also for the more restrictive case when ties occur only on one side of the bipartition, and any preference list is either strictly ordered or a tie of length two [24]. As SRTI-Existence is NP-hard for complete graphs, all three problems considered in this paper are NP-hard on complete graphs (as every stable matching is a maximal matching). This implies paraNP-hardness for all parameters which are constant on cliques, including distance to clique, cliquewidth, neighborhood diversity, the number of uncovered vertices, and modular width.

Following up on work by Bartholdi III and Trick [2], Bredereck et al. [3] showed NP-hardness and polynomial-time solvability results for SRTI-Existence under several restrictions constraining the agents' preference lists.

On a fairly general level, there is quite some work on employing methods of parameterized algorithmics in the search for potential islands of tractability for in general NP-hard stable matching problems [1, 5, 6, 26, 28]. More specifically, Marx and Schlotter [26] showed that Max-SRTI is W[1]-hard when parameterized by the number of ties. They observed that it is NP-hard even if the maximum length of a tie is constant but showed that Max-SRTI is fixed-parameter tractable when parameterized by the combined parameter "number of ties and maximum length of a tie". Meeks and Rastegari [30] considered a setting where the agents are partitioned into different types having the same preferences. They show that the problem is FPT in the number of types. Mnich and Schlotter [31] defined Stable Marriage with Covering Constraints, where the task is to find a matching which matches a given set of agents, and minimizes the number of blocking pairs among all these matchings. They showed the NP-hardness of this problem and investigated several parameters such as the number of blocking pairs or the maximum degree of the acceptability graph.

[1] In the following, we consider a slightly different formulation of these problems: We assume that the input consists of the acceptability graph and rank functions. This is no restriction, as one can transform a set of agents and a profile to an acceptability graph and rank functions and vice versa in linear time.

Most importantly for our work, however, Adil et al. [1] started the research on structural restrictions of the acceptability graph, which we continue and extend. Their result is an XP-algorithm for the parameter treewidth; indeed, they did not show W[1]-hardness for this parameter, leaving this as an open question. This open question was solved (also for the bipartite case) by Gupta et al. [15], who further considered various variants (such as sex-equal or balanced) of stable marriage with respect to two variants of treewidth.[2] Moreover, Adil et al. [1] showed that Max-SRTI is fixed-parameter tractable when parameterized by the size of the solution (that is, the cardinality of the set of edges in the stable matching) and that Max-SRTI restricted to planar acceptability graphs then is fixed-parameter tractable even with subexponential running time.[3]

Our Contributions. We continue the study of algorithms for Max-SRTI and its variants based on structural limitations of the acceptability graph. In particular, we extend the results of Adil et al. [1] in several ways. For an overview on our results we refer to Figure 1. We highlight a few results in what follows. We observe that Adil et al.'s dynamic programming-based XP-algorithm designed for the parameter treewidth[4] indeed yields fixed-parameter tractability for the combined parameter treewidth and maximum degree. We complement their XP result and the above mentioned results by showing that Max-SRTI is W[1]-hard for the graph parameters treedepth, tree-cut width, and feedback vertex set. Notably, all these graph parameters are "weaker" [22] than treewidth and these mutually independent results imply W[1]-hardness with respect to treewidth; the latter was also shown in the independent work of Gupta et al. [15].

For the two related problems Perfect-SRTI and SRTI-Existence, on the contrary we show fixed-parameter tractability with respect to the parameter tree-cut width. These results confirm the intuition that tree-cut width, a recently introduced [35] and since then already well researched graph parameter [11, 12, 13, 20, 27] "lying between" treewidth and the combined parameter "treewidth and maximum vertex degree", is a better suited structural parameter for edge-oriented problems than treewidth is. Moreover, we extend our W[1]-hardness results to Perfect-SRTI and SRTI-Existence parameterized by treedepth and to Perfect-SRTI parameterized by the feedback vertex number.

In summary, we provide a fairly complete picture of the (graph-)parameterized computational complexity landscape for the three studied problems – see Figure 1 for an overview of our results. Among other things Figure 1 for the parameter tree-cut width depicts a surprising complexity gap between Max-SRTI on the one side (W[1]-hardness) and Perfect-SRTI and SRTI-Existence (fixed-parameter tractability) on the other side. Finally, Figure 1 leaves as an open question the parameterized complexity of SRTI-Existence with respect to the parameter feedback vertex set number which we conjecture to be answered with W[1]-hardness.

---

[2] Indeed, without knowing the work of Gupta et al. [15] our work initially was strongly motivated by Adil et al.'s [1] open question for treewidth. To our surprise, although the Adil et al. [1] paper has been revised six months after the publication of Gupta et al. [15], it was not mentioned by Adil et al. [1] that this open question was answered by a subset of the authors, namely Gupta et al. [15].

[3] More precisely, Adil et al. state their result for the parameter "size of a maximum matching of the acceptability graph", which is only by a factor at most two greater than the size of a stable matching.

[4] It only gives containment in XP for this parameter, and only this is stated by Adil et al. [1].

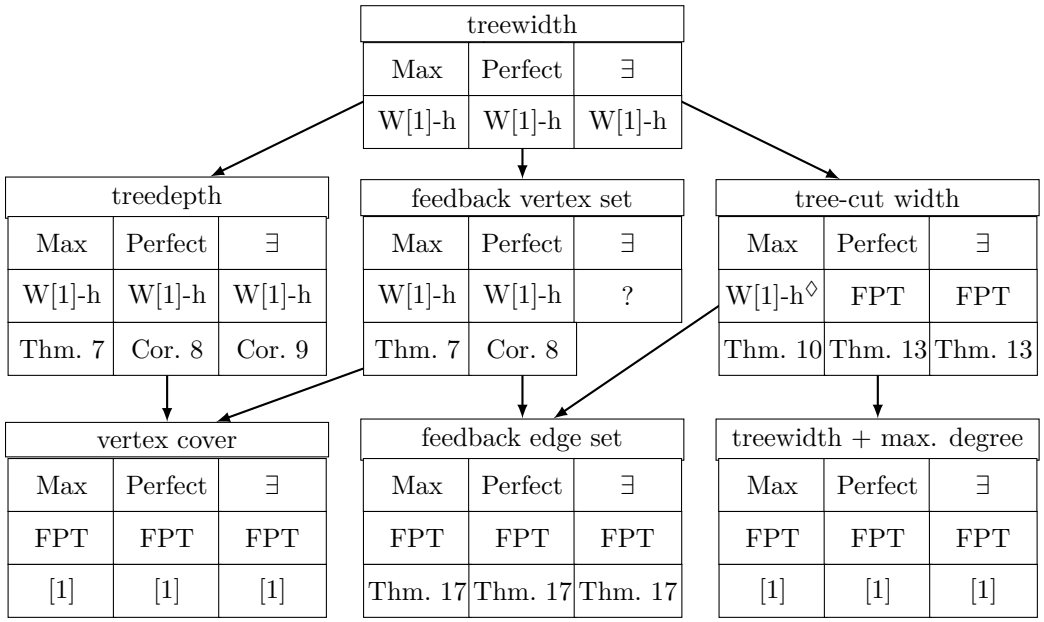

Figure 1 Results for graph-structural parameterizations of Stable Roommates with Ties and Incomplete Lists. Max means Max-SRTI, Perfect means Perfect-SRTI, and ∃ means SRTI-Existence. The symbol ◊ indicates the existence of an FPT factor-$\frac{1}{2}$-approximation algorithm (see Corollary 16). The arrows indicate dependencies between the different parameters. An arrow from a parameter p_1 to a parameter p_2 means that there is a computable function $f\colon \mathbb{N} \to \mathbb{N}$ such that for any graph G we have $p_1(G) \leq f(p_2(G))$. Consequently, fixed-parameter tractability for p_1 then implies fixed-parameter tractability for p_2, and W[1]-hardness for p_2 then implies W[1]-hardness for p_1.

2 Preliminaries

For a positive integer n let $[n] := \{1, 2, 3, \ldots, n\} = \{x \in \mathbb{N} : x \leq n\}$. We write vectors $\boldsymbol{h}$ in boldface, and access their entries (coordinates) via $\boldsymbol{h}(e)$.

For a graph G and a vertex $v \in V(G)$, let $\delta_G(v)$ be the set of edges incident to v. If the graph G is clear from the context, then we may just write $\delta(v)$. For a subset of edges $M \subseteq E(G)$ and a vertex $v \in V(G)$, we define $\delta_M(v) := \delta_G(v) \cap M$. We denote the maximum degree in G by $\Delta(G)$, i.e., $\Delta(G) := \max_{v \in V(G)} |\delta_G(v)|$. For a tree T rooted at a vertex r and a vertex $v \in V(T)$, we denote by T_v the subtree rooted at v. For a graph G and a subset of vertices X (a subset of edges F), we define $G - X$ ($G - F$) to be the graph arising from G by deleting all vertices in X and all edges incident to a vertex from X (deleting all edges in F). For a graph G and a set of vertices $X \subseteq V(G)$, the graph arising by *contracting* X is denoted by $G_{/X}$; it is defined by replacing the vertices in X by a single vertex. Thus, we have $V(G_{/X}) := \big(V(G) \setminus X\big) \cup \{v_X\}$ and $E(G_{/X}) := \{\{v, w\} \in E(G) : v, w \notin X\} \cup \{\{v, v_X\} : \{v, x\} \in E(G) : v \notin X, x \in X\}$. Unless stated otherwise, $n := |V(G)|$, and $m := |E(G)|$.

We define the directed graph $\overleftrightarrow{G}$ by replacing each edge $\{v, w\} \in E(G)$ by two directed ones in opposite directions, i.e. (v, w) and (w, v).

Note that the acceptability graph for a set of agents V and a profile $\mathcal{P}$ is always simple, while a graph arising from a simple graph through the contraction of vertices does not need to be simple.

A parameterized problem consists of the problem instance I (in our setting the Stable Roommate instance) and a parameter value k (in our case always a number measuring some aspect in acceptability graph). An FPT-algorithm for a parameterized problem is an algorithm that runs in time $f(k)|I|^{O(1)}$, where f is some computable function. That is, an FPT algorithm can run in exponential time, provided that the exponential part of the running time depends on the parameter value only. If such an algorithm exists, the parameterized problem is called fixed-parameter tractable for the corresponding parameter. There is also a theory of hardness of parameterized problems that includes the notion of W[1]-hardness. If a problem is W[1]-hard for a given parameter, then it is widely believed not to be fixed-parameter tractable for the same parameter.

The typical approach to showing that a certain parameterized problem is W[1]-hard is to reduce to it a known W[1]-hard problem, using the notion of a parameterized reduction. In our case, instead of using the full power of parameterized reductions, we use standard many-one reductions that ensure that the value of the parameter in the output instance is bounded by a function of the parameter of the input instance.

The Exponential-Time Hypothesis (ETH) of Impagliazzo and Paturi [17] asserts that there is a constant $c > 1$ such that there is no $c^{o(n)}$ time algorithm solving the Satisfiability problem, where n is the number of variables. Chen et al. [4] showed that assuming ETH, there is no $f(k) \cdot n^{o(k)}$ time algorithm solving k-(Multicolored) Clique, where f is any computable function and k is the size of the clique we are looking for. For further notions related to parameterized complexity and ETH refer to [8].

2.1 Profiles and preferences

Let V be a set of agents. A *preference list* $\mathcal{P}_v$ for an agent v is a subset $P_v \subseteq V \setminus \{v\}$ together with an ordered partition $(P_v^1, P_v^2, \ldots, P_v^k)$ of P_v. A set P_v^i with $|P_v^i| > 1$ is called a *tie*. The *size of a tie* P_v^i is its cardinality, i.e., $|P_v^i|$. For an agent $v \in V$, the *rank function* is $\mathrm{rk}_v \colon P_v \cup \{v\} \to \mathbb{N} \cup \{\infty\}$ with $\mathrm{rk}_v(x) := i$ for $x \in P_v^i$, and $\mathrm{rk}_v(v) = \infty$.

We say that v *prefers* $x \in P_v$ *over* $y \in P_v$ if $\mathrm{rk}_v(x) < \mathrm{rk}_v(y)$. If $\mathrm{rk}_v(x) = \mathrm{rk}_v(y)$, then v *ties* x and y. For a set V of agents, a set $\mathcal{P} = (\mathcal{P}_v)_{v \in V}$ of preference lists is called a *profile*. The corresponding *acceptability graph* G consists of vertex set $V(G) := V$ and edge set $E(G) := \{\{v, w\} : v \in P_w \wedge w \in P_v\}$.

A subset $M \subseteq E(G)$ of pairwise non-intersecting edges is called a matching. If $\{x, y\} \in M$, then we denote the corresponding partner y of x by $M(x)$ and set $M(x) := x$ if x is unmatched, that is, if $\{y \in V(G) : \{x, y\} \in M\} = \emptyset$. An edge $\{v, w\} \in E(G)$ is *blocking for* M if $\mathrm{rk}_v(w) < \mathrm{rk}_v(M(v))$ and $\mathrm{rk}_w(v) < \mathrm{rk}_w(M(w))$; we say that v, w constitutes a *blocking pair for* M. A matching $M \subseteq E(G)$ is *stable* if there are no blocking pairs, i.e., for all $\{v, w\} \in E(G)$, we have $\mathrm{rk}_v(w) \geq \mathrm{rk}_v(M(v))$ or $\mathrm{rk}_w(v) \geq \mathrm{rk}_v(M(w))$.

Note that the literature contains several different stability notions for a matching in the presence of ties. Our stability definition is frequently called *weak stability*.[5]

2.2 Structural graph parameters

We consider the (graph-theoretic) parameters treewidth, tree-cut width, treedepth, feedback vertex number, feedback edge number, vertex cover number, and the combined parameter "treewidth + maximum vertex degree" (also called degree-treewidth in the literature).

[5] Manlove [23] discusses other types of stability – strong stability and super-strong stability.

A set of vertices $S \subseteq V(G)$ is a *feedback vertex set* if $G - S$ is a forest and the *feedback edge set* is a subset $F \subseteq E(G)$ of edges such that $G - F$ is a forest. We define the *feedback vertex (edge) number* $\mathrm{fvs}(G)$ ($\mathrm{fes}(G)$) to be the cardinality of a minimum feedback vertex (edge) set of G. A vertex cover is a set of vertices intersecting with every edge of G, and the vertex cover number $\mathrm{vc}(G)$ is the size of a minimum vertex cover. The *treedepth* $\mathrm{td}(G)$ is the smallest height of a rooted tree T with vertex set $V(G)$ such that for each $\{v, w\} \in V(G)$ we have that either v is a descendant of w in T or w is a descendant of v in T.

Treewidth intuitively measures the tree-likeness of a graph. It can be defined via structural decompositions of a graph into pieces of bounded size, which are connected in a tree-like fashion, called *tree decompositions*.

Tree-Cut Width. Tree-cut width has been introduced by Wollan [35] as tree-likeness measure between treewidth and treewidth combined with maximum degree. A family of subsets $X_1, \ldots, X_k$ of a finite set X is a *near-partition* of X if $X_i \cap X_j = \emptyset$ for all $i \neq j$ and $\bigcup_{i=1}^{k} X_i = X$. Note that $X_i = \emptyset$ is possible (even for several distinct i). A *tree-cut decomposition* of a graph G is a pair $(T, \mathcal{X})$ which consists of a tree T and a near-partition $\mathcal{X} = \{X_t \subseteq V(G) : t \in V(T)\}$ of $V(G)$. A set in the family $\mathcal{X}$ is called a *bag* of the tree-cut decomposition. Given a tree node t, let T_t be the subtree of T rooted at t. For a node $t \in V(T)$, we denote by Y_t the set of vertices induced by T_t, i.e. $Y_t := \bigcup_{v \in V(T_t)} X_v$.

For an edge $e = \{u, v\} \in E(T)$, we denote by $T_u^{\{u,v\}}$ and $T_v^{\{u,v\}}$ the two connected components in $T - e$ which contain u respectively v. These define a partition

$$\Big(\bigcup_{t \in T_u^{\{u,v\}}} X_t, \bigcup_{t \in T_v^{\{u,v\}}} X_t \Big)$$

of $V(G)$. We denote by $\mathrm{cut}(e) \subseteq E(G)$ the set of edges of G with one endpoint in $\bigcup_{t \in T_u^{\{u,v\}}} X_t$ and the other one in $\bigcup_{t \in T_v^{\{u,v\}}} X_t$.

A tree-cut decomposition is called *rooted* if one of its nodes is called the root r. For any node $t \in V(T) \setminus \{r\}$, we denote by $e(t)$ the unique edge incident to t on the r-t-path in T. The *adhesion* $\mathrm{adh}_T(t)$ is defined as $|\mathrm{cut}(e(t))|$ for each $t \neq r$, and $\mathrm{adh}_T(r) := 0$.

The *torso of a tree-cut decomposition* $(T, \mathcal{X})$ *at a node* t, denoted by H_t, can be constructed from G as follows: If T consists of a single node, then the torso of $t \in V(T)$ is G. Else let $C_t^1, \ldots, C_t^\ell$ be the connected components of $T - t$. Let $Z_i := \bigcup_{v \in V(C_t^i)} X_v$. Then, the torso arises from G by contracting each $Z_i \subseteq V(G)$ for $1 \leq i \leq \ell$.

The operation of *suppressing a vertex* v of degree at most two consists of deleting v and, if v has degree exactly two, then adding an edge between the two neighbors of v. The torso-size $\mathrm{tor}(t)$ is defined as the number of vertices of the graph arising from the torso H_t by exhaustively suppressing all vertices of degree at most two.

The *width of a tree-cut decomposition* $(T, \mathcal{X})$ is defined as $\max_{t \in V(T)} \{\mathrm{adh}(t), \mathrm{tor}(t)\}$. The *tree-cut width* $\mathrm{tcw}(G)$ of a graph G is the minimum width of a tree-cut decomposition of G.

Nice tree-cut decompositions. Similarly to nice tree decompositions [21], each tree-cut decomposition can be transformed into a nice tree-cut decomposition. Nice tree-cut decompositions have additional properties which help simplifying algorithm design. Besides the definition of nice tree-cut decompositions, in the following we provide some of its properties.[6]

[6] The properties used here are stated (without a proof) by Ganian et al. [11].

▶ **Definition 1** ([11]). *Let $(T, \mathcal{X})$ be a tree-cut decomposition. A node $t \in V(T)$ is called* light *if* $\mathrm{adh}(t) \leq 2$ *and all outgoing edges from Y_t end in X_p, where p is the parent of t, and* heavy *otherwise (see Figure 2 for an example).*

▶ **Theorem 2** ([11, Theorem 2]). *Let G be a graph with $\mathrm{tcw}(G) = k$. Given a tree-cut decomposition of G of width k, one can compute a nice tree-cut decomposition $(T, \mathcal{X})$ of G of width k with at most $2|V(G)|$ nodes in cubic time.*

▶ **Lemma 3** ([11, Lemma 2]). *Each node t in a nice tree-cut decomposition of width k has at most $2k + 1$ heavy children.*

In what follows, we will assume that a nice tree-cut decomposition of the input graph is given. Computing the tree-cut width of a graph is NP-hard [20], but there exists an algorithm, given a graph G and an integer k, either finds a tree-cut decomposition of width at most $2k$ or decides that $\mathrm{tcw}(G) > k$ in time $2^{O(k^2 \log k)} n^2$. Furthermore, Giannopoulou et al. [14] gave a constructive proof of the existence of an algorithm deciding whether the tree-cut width of a given graph G is at most k in $f(k)n$ time, where f is a computable recursive function. Very recently, Ganian et al. [13] performed experiments on computing optimal tree-cut decompositions using SAT-solvers.

▶ **Lemma 4.** *Let T be a forest. Then* $\mathrm{tcw}(T) = 1$.

Proof. As clearly $\mathrm{tcw}(T) \leq \mathrm{tcw}(T + F)$ for any set of edges F, we may assume without loss of generality that T is a tree.

We define $X_t = \{t\}$ for all $t \in V(T)$, and consider the tree-cut decomposition $(T, \mathcal{X})$, and pick an arbitrary vertex r to be the root of T.

As T is a tree, we have $\mathrm{adh}(t) = 1$ for all $t \neq r$.

Furthermore, for each $t \in V(T)$, all vertices but t contained in the torso of t can be suppressed, and thus $\mathrm{tor}(t) \leq 1$. ◀

▶ **Lemma 5.** *Let G be a graph. Then* $\mathrm{tcw}(G + e) \leq \mathrm{tcw}(G) + 2$ *for any edge e.*

Proof. Consider a tree-cut decomposition $(T, \mathcal{X})$ of G. This is also a tree-cut decomposition of $G + e$.

Clearly, the adhesion of any node of T can increase by at most 1.

The torso-size of a vertex can also increase by at most 2, as e can prevent at most both of its endpoints from being suppressed. ◀

▶ **Corollary 6.** *Let G be a graph, and k the feedback edge number. Then* $\mathrm{tcw}(G) \leq 2k + 1$.

Proof. This directly follows from Lemmas 4 and 5. ◀

3 Hardness Results

All our hardness result are based on parameterized reductions from the Multicolored Clique problem, a well-known W[1]-hard problem. The so-called vertex selection gadgets are somewhat similar to those of Gupta et al. [15], however, the other gadgets in our reductions are different. Here we only discuss the main dissimilarities of the reductions we present here and the one of Gupta et al. [15]. We use one gadget for each edge whereas the reduction presented therein uses a single gadget for all edges between two color classes. This subtle difference allows us to bound not only treewidth of the resulting graph but rather both treedepth and the size of a feedback vertex set. It is worth noting that it is not clear whether the reduction of Gupta et al. [15] can, with some additional changes and work, yield hardness

for these parameters as well or not. On the other hand, we use some consistency gadget which is essentially a triangle (while the graph resulting from the reduction of Gupta et al. [15] is bipartite). Furthermore, in our reduction all of the vertices have either strictly ordered preferences or a tie between (the only) two agents they find acceptable. We defer details and further comments to a full version of this paper due to space reasons.

▶ **Theorem 7** (⋆). Max-SRTI *parameterized by treedepth and feedback vertex set is W[1]-hard. Moreover, there is no* $n^{o(\mathrm{td}(G))}$ *time algorithm for* Max-SRTI*, unless ETH fails.*

Note that such a maximum stable matching corresponding to a clique of size k leaves only $2k(k-1)$ vertices uncovered. Thus, by adding $2k(k-1)$ vertices connected to all other vertices, we also get the W[1]-hardness for Perfect-SRTI:

▶ **Corollary 8** (⋆). Perfect-SRTI *parameterized by treedepth and feedback vertex set is W[1]-hard.*

From this, we get the W[1]-hardness of SRTI-Existence for treedepth by adding for each vertex a 3-cycle, ensuring that this vertex is matched (similarly to the 3-cycles c_{ij}, c'_{ij}, c''_{ij} for the vertex c_{ij} in the consistency gadgets).

▶ **Corollary 9** (⋆). SRTI-Existence *parameterized by treedepth is W[1]-hard.*

A different reduction partially using similar ideas and techniques yields the W[1]-hardness of Max-SRTI for the parameter tree-cut width:

▶ **Theorem 10** (⋆). Max-SRTI *parameterized by tree-cut width is W[1]-hard.*

4 Tractability Results

We present an FPT-algorithm for Perfect-SRTI and SRTI-Existence. Given a tree-cut decomposition of the acceptability graph, we use dynamic programming to decide whether a solution exists or not. In the dynamic programming table for a node t we store information whether there exists a matching M for the set Y_t of vertices from the bags of the subtree of the tree-cut decomposition rooted in t. We allow that M is not stable in G but require that the blocking pairs are incident to vertices outside Y_t, and for some of the edges $\{v, w\}$ in $\mathrm{cut}(t)$, we require the endpoint v in Y_t to be matched at least as good as he ranks w.

DP Tables. Before we describe the idea behind the table entries we store in our dynamic programming procedure, we introduce the following relaxation of matching stability.

▶ **Definition 11.** *Let* $(T, \mathcal{X})$ *be a nice tree-cut decomposition of* G*. For a node* $t \in V(T)$*, the* closure of t *(*$\mathrm{clos}(t)$*) is the set of vertices in* Y_t *together with their neighbors, that is,* $\mathrm{clos}(t) := Y_t \cup N_G(Y_t)$ *. We say that a matching* M *on* $\mathrm{clos}(t)$ *for some* $t \in V(T)$ complies with a vector $\boldsymbol{h} \in \{-1, 0, 1\}^{\mathrm{cut}(t)}$ *if the following conditions hold:*

- *for each edge* $e \in \mathrm{cut}(t)$*, we have* $e \in M$ *if and only if* $\boldsymbol{h}_e = 0$*;*
- *for each* $e = \{v, w\} \in \mathrm{cut}(t)$ *with* $v \in Y_t$ *and* $\boldsymbol{h}_e = 1$*, we have* $\mathrm{rk}_v(M(v)) \leq \mathrm{rk}_v(w)$*, i.e.* v *ranks its partner (not being* w *by the previous condition) in* M *at least as good as* w*; and*
- *every blocking pair contains a vertex from* $V(G) \setminus Y_t$ *not matched in* M*.*

Intuitively, if we set $\boldsymbol{h}_t(e) = 1$ for an edge $e = \{v, x\}$ in $\mathrm{cut}(t)$ with $x \in X_t$, then we are searching for a matching M (in $G[\mathrm{clos}(t)]$) for which we can guarantee that $\mathrm{rk}_x(M(x)) \leq \mathrm{rk}_x(v)$. Consequently, we know that e will not be blocking in an extension of such a matching. Contrary, if we set $\boldsymbol{h}_t(e) = -1$, then we allow x to prefer v over its partner (in particular, x

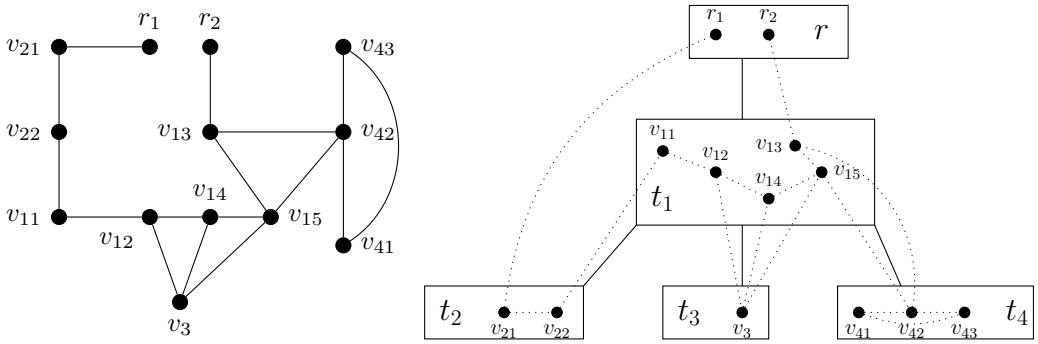

Figure 2 An example of a graph G (left part of the picture) and its nice tree-cut decomposition $(T, \mathcal{X})$ (not of minimal width). The vertices of G are the circles, while the nodes of T are the rectangles. For a node $t \in V(T)$, the bag X_t contains exactly those vertices inside the rectangle. In the right picture, the solid edges are the edges of T, while the dotted edges are from G. The nodes t_1 and t_4 are light, while t_2 (because there is an edge connecting a vertex in t_2 to a vertex in r) and t_3 (because $\mathrm{adh}(t_3) = 3$) are heavy.

may remain unmatched). Thus, for an extension of such a matching in order to maintain stability we have to secure that $\mathrm{rk}_v(M(v)) \leq \mathrm{rk}_v(x)$, since otherwise e will be blocking. Observe that if a matching complies with $\boldsymbol{h}$ for a vector $\boldsymbol{h} \in \{-1, 0, 1\}^{\mathrm{cut}(t)}$ with $\boldsymbol{h}(e) = 1$ for some edge $e \in \mathrm{cut}(t)$, then it complies with $\hat{\boldsymbol{h}} \in \{-1, 0, 1\}^{\mathrm{cut}(t)}$ which is the same as $\boldsymbol{h}$ but for e is set to -1 (formally, $\hat{\boldsymbol{h}}(f) = \boldsymbol{h}(f)$ for $f \neq e$ and $\hat{\boldsymbol{h}}(e) = -1$). Clearly, any matching complying with $\boldsymbol{h}$ complies with $\hat{\boldsymbol{h}}$, since the latter is more permissive.

For a node $t \in T$ its dynamic programming table is τ_t and it contains an entry for every $\boldsymbol{h} \in \{-1, 0, 1\}^{\mathrm{cut}(t)}$. An entry $\tau_t[\boldsymbol{h}]$ is a matching $M \subseteq E(G[Y_t]) \cup \mathrm{cut}(t)$ if M complies with $\boldsymbol{h}$. If no such matching for $\boldsymbol{h}$ exists, then we set $\tau_t[\boldsymbol{h}] = \emptyset$. Note that the size of the table τ_t for a node t is upper-bounded by $3^{\mathrm{tcw}(G)}$.

▶ **Example 12.** The graph and tree-cut decomposition are depicted in Figure 2:

For t_1 and $\boldsymbol{h^1}(\{t_1, v_{21}\}) = 0$ and $\boldsymbol{h^1}(\{v_{12}, r_2\}) = 1$, all stable matchings containing $\{t_1, v_{21}\}$ and $\{v_{21}, v_{22}\}$ are complying with $\boldsymbol{h}$. For t_2 and $\boldsymbol{h^2}(\{v_{21}, r_1\}) = 1$ and $\boldsymbol{h^2}(\{v_{22}, v_{11}\}) = -1$, the matching $M = \{\{v_{21}, v_{22}\}\}$ is complying with h^2. For t_3 and any vector $\boldsymbol{h^3}$ with $\boldsymbol{h^3}(\{v_3, v_{12}\}) = 1$, no matching complies with $\boldsymbol{h^3}$: In such a matching, v_3 must be ranked at least as good as $\mathrm{rk}_{v_3}(v_{12}) = 1$, but not to v_{12}, which is impossible. For t_4 and $\boldsymbol{h^4}(\{v_{42}, v_{15}\}) = 0 = \boldsymbol{h^4}(\{v_{42}, v_{13}\})$, no matching complying with $\boldsymbol{h^4}$ exists, as such a matching must match v_{42} to both v_{13} and v_{15}.

▶ **Theorem 13** ($\star$). Perfect-SRTI *and* SRTI-Existence *can be solved in* $2^{O(k \log k)} n^{O(1)}$ *time, where* $k := \mathrm{tcw}(G)$.

Proof Sketch. Let $(T, \mathcal{X})$ be a nice tree-cut decomposition of G of width k. We will first explain the algorithm for SRTI-Existence, and in the end we highlight how this algorithm can be adapted to Perfect-SRTI. We compute the values $\tau_t[\boldsymbol{h}]$ by bottom-up induction over the tree T.

For a leaf $t \in V(T)$ and a vector $\boldsymbol{h}$, we enumerate all matchings M_t on $G[X_t \cup N(Y_t)]$. We check whether M_t complies with $\boldsymbol{h}$. If we find such a matching, then we store one of these matchings in $\tau_t[\boldsymbol{h}]$, and else set $\tau_t[\boldsymbol{h}] = \emptyset$. As $|X_t \cup N(Y_t)| \leq 2k$, and G is simple, the number of matchings is bounded by $2^{O(k \log k)}$.

The induction step, that is, computing the table entries for the inner nodes of the tree-cut decomposition is the most-involved part and sketched below.

For the root $r \in V(T)$, we have $Y_r = V(G)$ and $\mathrm{cut}(r) = \emptyset$. Thus, a matching on $Y_r = V(G)$ complying with $\boldsymbol{h^r} \in \{-1,0,1\}^{\emptyset}$ is just a stable matching (note that $\boldsymbol{h^r}$ is unique). Hence, G contains a stable matching if and only if $\tau_r[\boldsymbol{h^r}] \neq \emptyset$.

The induction step is executed for each $t \in V(T)$ and each $\boldsymbol{h} \in \{-1,0,1\}^{\mathrm{cut}(t)}$, and therefore at most $n3^k$ times. As each execution takes $2^{O(k \log k)} n^{O(1)}$ time, the total running time of the algorithm is bounded by $2^{O(k \log k)} n^{O(1)}$.

To solve Perfect-SRTI, we store in any dynamic programming table τ_t only matchings such that every vertex inside Y_t is matched.

Induction Step. In what follows we sketch how to solve the induction step.

Induction Step

Input: The acceptability graph G, rank functions rk_v for all $v \in V(G)$, a tree-cut decomposition $(T, \mathcal{X})$, a node $t \in V(T)$, a vector $\boldsymbol{h} \in \{-1,0,1\}^{\mathrm{cut}(t)}$, for each child c of t and each $\boldsymbol{h^c} \in \{-1,0,1\}^{\mathrm{cut}(c)}$ the value $\tau_c[\boldsymbol{h^c}]$.

Task: Compute $\tau_t[\boldsymbol{h}]$.

Before we give the proof idea we first give the definition of light children classes. Intuitively, two light children of a note t are in the same class if, with respect to t, they behave in a similar way, that is, their neighborhood in X_t is the same and their table entries are compatible. In order to properly define the later notion we first need to introduce few auxiliary definitions. To simplify the notation, we assume that the edges of G are enumerated, that is, we have $E(G) = \{e_1, e_2, \ldots, e_m\}$. For a vector $\boldsymbol{h} \in \{-1,0,1\}^{\mathrm{cut}(t)}$, the i-th coordinate will always be the coordinate of the edge with the i-th lowest index in $\mathrm{cut}(t)$.

▶ **Definition 14.** *Let $t \in V(T)$ be a node. We define the* signature $\mathrm{sig}(t)$ *to be the set* $\{\boldsymbol{h} \in \{-1,0,1\}^{\mathrm{cut}(t)} : \tau_t[\boldsymbol{h}] \neq \emptyset\}$.

Let c, d be light children of t. We write $c \Diamond d$ if and only if $\mathrm{sig}(c) = \mathrm{sig}(d)$ *and* $N(c) = N(d)$, *where we define $N(c) := N_G(Y_c)$ for each $c \in V(T)$.*

It follows immediately that $\Diamond$ is an equivalence relation on light children of t. Furthermore, since each class of $\Diamond$ is identified by its signature and neighborhood in X_t, there are $O(k^2)$ classes of $\Diamond$. Let $\mathcal{C}(c)$ denote the equivalence class of the light child c of t and let $N(\mathcal{C}) \subseteq X_t$ be the set of neighbors of the class $\mathcal{C}$ of $\Diamond$ (i.e., $N(\mathcal{C})$ is the set of neighbors $N(Y_c) \subseteq X_t$ for $c \in \mathcal{C}$). Furthermore, let $\mathrm{sig}(\mathcal{C})$ denote the signature of the class $\mathcal{C}$ and similarly let $\mathrm{sig}_x(\mathcal{C})$ denote the signature of $\mathcal{C}$ with respect to its neighbor $x \in N(\mathcal{C})$.

▶ **Observation 15.** *If $\mathcal{C}$ is a class with $|\mathcal{C}| \geq 3$ and $(-1,-1) \notin \mathrm{sig}(\mathcal{C})$, then there is no stable matching in G.*

Proof Idea (Induction Step). Due to space reasons, we defer proof details to a full version of this paper. First, we will guess which edges incident to heavy children are in the matching M to be computed and which are not. Note that there are at most $k(2k+1)$ edges incident to heavy children of t, since their adhesion is at most k. Thus, we fix a matching between vertices in X_t and heavy children and what remains is to combine the guessed matching with matchings in their graphs; note that we can also guess these, however, this results in $(3^k)^{O(k)}$ guesses. Instead of trying all of the possibilities we prove that it is possible to reduce the number of heavy children matchings we try to extend to $k^{O(k)}$. It is worth noting that these choices will result in some further constraints the matching in the light children must fulfill.

Then for every class of the equivalence $\Diamond$ we guess whether its neighbor(s) in X_t are matched to it (i.e., matched to a vertex in a child or two in this class) or not. Let $\mathcal{N}$ denote the guessed matching. Note that there are $k^{O(k)}$ such choices, since each vertex $v \in X_t$ is "adjacent" to at most $k+1$ classes of $\Diamond$ (i.e., there are at most k classes such that v is adjacent to a vertex in all of the children contained in this class) and choosing a class (or deciding not to be adjacent to any light child) for every vertex in X_t yields the claimed bound. We show that if a class of $\Diamond$ with $N(\mathcal{C}) = \{x\}$ is selected to be matched with its neighbor x, then it is possible to match x to the best child in this class (the one containing the top choice for x among these children); provided there exists a solution which is compatible with such a choice. We do this by showing a rather simple exchange argument.

Having resolved heavy children and light children with only one neighbor in X_t it remains to deal with children with two neighbors. We generalise the exchange argument we provide for classes with one neighbor. Then, we prove that many combinations of $\mathcal{N}$ and a signature of a class $\mathcal{C}$ allows us to reduce the number of children in $\mathcal{C}$ in which we have to search for a partner of a vertex in $N(\mathcal{C})$ to a constant (in fact, four). We call such classes the *good classes*. However, there are classes where this is not possible (call these the *bad classes*). Consequently, there are only 4^k possibilities how to match vertices in X_t to good classes of children. Finally, we characterise the bad classes and use this characterisation to show how to reduce the question of existence of a (perfect) matching complying with $\boldsymbol{h}$ and obeying all the constraints of heavy children to an instance of 2-SAT (similarly to Feder [9]). ◀

▶ **Corollary 16** (⋆). *A factor-$\frac{1}{2}$-approximation for* Max-SRTI *can be computed in $2^{O(k \log k)} n^{O(1)}$ time, where $k := \mathrm{tcw}(G)$.*

Using standard techniques and the polynomial-time algorithm for graphs of bounded treewidth by Adil et al. [1], we obtain an FPT-algorithm for Max-SRTI (and therefore also Perfect-SRTI and SRTI-Existence) parameterized by feedback edge number:

▶ **Theorem 17** (⋆). Max-SRTI *can be solved in $2^{\mathrm{fes}(G)} n^{O(1)}$ time.*

5 Conclusion

Taking the viewpoint of parameterized graph algorithmics, we investigated the line between fixed-parameter tractability and W[1]-hardness of Stable Roommates with Ties and Incomplete Lists. Studying parameterizations mostly relating to the "tree-likeness" of the underlying acceptability graph, we arrived at a fairly complete picture (refer to Figure 1) of the corresponding parameterized complexity landscape. There is a number of future research issues stimulated by our work. First, we did not touch on questions about polynomial kernelizability of the fixed-parameter tractable cases. Indeed, for the parameter feedback edge number we believe that a polynomial kernel should be possible. Another issue is how tight the running time for our fixed-parameter algorithm for the parameter tree-cut width k is; more specifically, can we show that our exponential factor $k^{O(k)}$ is basically optimal or can it be improved to say $2^{O(k)}$? Also the case of SRTI-Existence parameterized by feedback vertex number remained open (see Figure 1). Based on preliminary considerations, we conjecture it to be W[1]-hard. Clearly, there is still a lot of room to study Stable Roommates with Ties and Incomplete Lists through the lens of further graph parameters. On a general note, we emphasize that so far our investigations are on the purely theoretic and classification side; practical algorithmic considerations are left open for future research.

References

1 Deeksha Adil, Sushmita Gupta, Sanjukta Roy, Saket Saurabh, and Meirav Zehavi. Parameterized algorithms for stable matching with ties and incomplete lists. *Theor. Comput. Sci.*, 723:1–10, 2018.

2 John J. Bartholdi III and Michael Trick. Stable Matching with Preferences Derived from a Psychological Model. *Oper. Res. Lett.*, 5(4):165–169, 1986.

3 Robert Bredereck, Jiehua Chen, Ugo Paavo Finnendahl, and Rolf Niedermeier. Stable Roommate with Narcissistic, Single-Peaked, and Single-Crossing Preferences. In *Proc. of ADT '17*, volume 10576 of *LNCS*, pages 315–330. Springer, 2017.

4 Jianer Chen, Benny Chor, Mike Fellows, Xiuzhen Huang, David Juedes, Iyad A. Kanj, and Ge Xia. Tight lower bounds for certain parameterized NP-hard problems. *Information and Computation*, 201(2):216–231, 2005.

5 Jiehua Chen, Danny Hermelin, Manuel Sorge, and Harel Yedidsion. How Hard Is It to Satisfy (Almost) All Roommates? In *Proc. of ICALP '18*, volume 107 of *LIPIcs*, pages 35:1–35:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

6 Jiehua Chen, Rolf Niedermeier, and Piotr Skowron. Stable Marriage with Multi-Modal Preferences. In *Proc. of EC '18*, pages 269–286. ACM, 2018.

7 Ágnes Cseh, Robert W. Irving, and David F. Manlove. The Stable Roommates Problem with Short Lists. *Theory Comput. Syst.*, 63(1):128–149, 2019.

8 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

9 Tomás Feder. A new fixed point approach for stable networks and stable marriages. *J. Comput. Syst. Sci.*, 45(2):233–284, 1992.

10 Anh-Tuan Gai, Dmitry Lebedev, Fabien Mathieu, Fabien de Montgolfier, Julien Reynier, and Laurent Viennot. Acyclic Preference Systems in P2P Networks. In *Proc. of Euro-Par '07*, volume 4641 of *LNCS*, pages 825–834. Springer, 2007.

11 Robert Ganian, Eun Jung Kim, and Stefan Szeider. Algorithmic Applications of Tree-Cut Width. In *Proc. of MFCS '15*, volume 9235 of *LNCS*, pages 348–360. Springer, 2015.

12 Robert Ganian, Fabian Klute, and Sebastian Ordyniak. On Structural Parameterizations of the Bounded-Degree Vertex Deletion Problem. In *Proc. of STACS '18*, volume 96 of *LIPIcs*, pages 33:1–33:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

13 Robert Ganian, Neha Lodha, Sebastian Ordyniak, and Stefan Szeider. SAT-encodings for treecut width and treedepth. In *Proc. of ALENEX '19*, pages 117–129. SIAM, 2019.

14 Archontia C. Giannopoulou, O-joung Kwon, Jean-Florent Raymond, and Dimitrios M. Thilikos. Lean Tree-Cut Decompositions: Obstructions and Algorithms. In *Proc. of STACS '19*, volume 126 of *LIPIcs*, pages 32:1–32:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2019.

15 Sushmita Gupta, Saket Saurabh, and Meirav Zehavi. On Treewidth and Stable Marriage. *CoRR*, abs/1707.05404, 2017. arXiv:1707.05404.

16 Dan Gusfield and Robert W. Irving. *The Stable Marriage Problem - Structure and Algorithms*. MIT Press, 1989.

17 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

18 Robert W. Irving. An Efficient Algorithm for the "Stable Roommates" Problem. *J. Algorithms*, 6(4):577–595, 1985.

19 Kazuo Iwama, Shuichi Miyazaki, Yasufumi Morita, and David Manlove. Stable Marriage with Incomplete Lists and Ties. In *Proc. of ICALP '99*, volume 1644 of *LNCS*, pages 443–452. Springer, 1999.

20 Eun Jung Kim, Sang-il Oum, Christophe Paul, Ignasi Sau, and Dimitrios M. Thilikos. An FPT 2-Approximation for Tree-Cut Decomposition. *Algorithmica*, 80(1):116–135, 2018.

21 Ton Kloks. *Treewidth, Computations and Approximations*, volume 842 of *LNCS*. Springer, 1994. doi:10.1007/BFb0045375.

22 Christian Komusiewicz and Rolf Niedermeier. New Races in Parameterized Algorithmics. In *Proc. of MFCS '12*, volume 7464 of *LNCS*, pages 19–30. Springer, 2012.

23 David Manlove. The structure of stable marriage with indifference. *Discrete Appl. Math.*, 122(1-3):167–181, 2002.

24 David Manlove, Robert W. Irving, Kazuo Iwama, Shuichi Miyazaki, and Yasufumi Morita. Hard variants of stable marriage. *Theor. Comput. Sci.*, 276(1-2):261–279, 2002.

25 David F. Manlove. *Algorithmics of Matching Under Preferences*, volume 2 of *Series on Theoretical Computer Science*. WorldScientific, 2013.

26 Dániel Marx and Ildikó Schlotter. Parameterized Complexity and Local Search Approaches for the Stable Marriage Problem with Ties. *Algorithmica*, 58(1):170–187, 2010.

27 Dániel Marx and Paul Wollan. Immersions in Highly Edge Connected Graphs. *SIAM J. Discrete Math.*, 28(1):503–520, 2014.

28 Dániel Marx and Ildikó Schlotter. Stable assignment with couples: Parameterized complexity and local search. *Discrete Optim.*, 8(1):25–40, 2011.

29 Eric McDermid. A 3/2-Approximation Algorithm for General Stable Marriage. In *Proc. of ICALP '09*, pages 689–700. Springer, 2009.

30 Kitty Meeks and Baharak Rastegari. Stable Marriage with Groups of Similar Agents. In *Proc. of WINE '18*, volume 11316 of *LNCS*, pages 312–326. Springer, 2018. doi:10.1007/978-3-030-04612-5_21.

31 Matthias Mnich and Ildikó Schlotter. Stable Marriage with Covering Constraints-A Complete Computational Trichotomy. In *Proc. of SAGT '17*, volume 10504 of *LNCS*, pages 320–332. Springer, 2017. doi:10.1007/978-3-319-66700-3_25.

32 Eytan Ronn. NP-complete stable matching problems. *J. Algorithms*, 11(2):285–304, 1990.

33 Alvin E. Roth, Tayfun Sönmez, and M. Utku Ünver. Pairwise kidney exchange. *J. Econ. Theory*, 125(2):151–188, 2005.

34 Alvin E. Roth, Tayfun Sönmez, and M. Utku Ünver. Efficient Kidney Exchange: Coincidence of Wants in Markets with Compatibility-Based Preferences. *Am. Econ. Rev.*, 97(3):828–851, June 2007.

35 Paul Wollan. The structure of graphs not admitting a fixed immersion. *J. Comb. Theory, Ser. B*, 110:47–66, 2015. doi:10.1016/j.jctb.2014.07.003.

36 H. Yanagisawa. *Approximation Algorithms for Stable Marriage Problems*. PhD thesis, Kyoto University, 2007.