Tight Approximations for Graphical House Allocation

Hadi Hosseini
Penn State University
State College, USA
hadi@psu.edu

Andrew McGregor
UMass Amherst
Amherst, USA
mcgregor@cs.umass.edu

Rik Sengupta
UMass Amherst
Amherst, USA
rsengupta@cs.umass.edu

Rohit Vaish
IIT Delhi
Delhi, India
rvaish@iitd.ac.in

Vignesh Viswanathan
UMass Amherst
Amherst, USA
vviswanathan@cs.umass.edu

ABSTRACT

The Graphical House Allocation problem asks: how can n houses (each with a fixed non-negative value) be assigned to the n vertices of an undirected graph G, so as to minimize the "aggregate local envy", i.e., the sum of absolute differences along the edges of G? This problem generalizes the classical Minimum Linear Arrangement problem, as well as the well-known *House Allocation Problem* from Economics, the latter of which has notable practical applications in organ exchanges. Recent work has studied the computational aspects of Graphical House Allocation and observed that the problem is NP-hard and inapproximable even on particularly simple classes of graphs, such as vertex disjoint unions of paths. However, the dependence of any approximations on the structural properties of the underlying graph had not been studied.

In this work, we give a complete characterization of the approximability of Graphical House Allocation. We present algorithms to approximate the optimal envy on general graphs, trees, planar graphs, bounded-degree graphs, bounded-degree planar graphs, and bounded-degree trees. For each of these graph classes, we then prove *matching* lower bounds, showing that in each case, no significant improvement can be attained unless P = NP. We also present general approximation ratios as a function of structural parameters of the underlying graph, such as treewidth; these match the aforementioned tight upper bounds in general, and are significantly better approximations for many natural subclasses of graphs. Finally, we present constant factor approximation schemes for the special classes of complete binary trees and random graphs.

Some of the technical highlights of our work are the use of expansion properties of Ramanujan graphs in the context of a classical resource allocation problem, and approximating optimal cuts in binary trees by analyzing the behavior of consecutive runs in bitstrings.

KEYWORDS

House Allocation; Envy Minimization; Local Envy

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

ACM Reference Format:
Hadi Hosseini, Andrew McGregor, Rik Sengupta, Rohit Vaish, and Vignesh Viswanathan. 2024. Tight Approximations for Graphical House Allocation. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

1 INTRODUCTION

In the EconCS community, the *House Allocation Problem* has been a topic of significant interest for some time [3, 16, 22, 29, 30]. In its canonical form, the problem involves a set of n agents, a set of n items ("houses"), and possibly different valuation functions for each agent. In general, given this framework, the problem asks for an "optimally fair" allocation of the houses to the agents. For instance, we might wish to minimize the total envy, or maximize the number of envy-free pairs of agents. In this context, as it is common in the fairness literature, an agent i envies an agent j in a particular allocation if according to agent i's valuation function, the item received by agent j is worth more than the item received by agent i; the amount of envy is the difference in these two values. The canonical problem has been studied in a variety of contexts, and is well-known as an algorithmically difficult problem to solve, for most reasonable fairness objectives.

Hosseini et al. [20] introduced a variant of the house allocation problem called Graphical House Allocation. In this setting, there are n agents, but now they are placed on the vertices of an undirected n-vertex graph $G = (V, E)$. There are still n items with arbitrary values, but the agents are *identical* in how they value these n items (i.e., they all agree on the value of each house). Graphical House Allocation now asks: how do we allocate each house to an agent so as to minimize the total envy along the edges of G?

We remark here that the setting where the agents are on a graph and only the envy along the graph edges is considered was studied before as well by Beynier et al. [3], who considered ordinal preferences in such a setting, and were interested in maximizing the number of envy-free edges in the underlying graph.

Observe that Graphical House Allocation is a purely combinatorial problem: we are given an n-vertex graph $G = (V, E)$ and a multiset $H = \{h_1, \ldots, h_n\} \subseteq \mathbb{R}_{\geq 0}$. We wish to find the bijective function $\pi : V \to H$ that minimizes $\sum_{(x,y)\in E} |\pi(x) - \pi(y)|$.

If the set of values were $H = \{1, \ldots, n\}$, then Graphical House Allocation would be identical to the well-known Minimum Linear Arrangement problem. This was observed by Hosseini et al. [20], who went on to show some remarkable differences between the two problems. For instance, while all hardness results carry over from Minimum Linear Arrangement to Graphical House Allocation, the latter is actually a significantly harder problem even on very simple graphs. In particular, in Minimum Linear Arrangement, we can assume without loss of generality that the underlying graph is connected; this is because an optimal solution is given by taking each connected component separately, and optimally assigning a contiguous subset of values to it. We lose this guarantee in Graphical House Allocation, even for small graphs with just two connected components. As a typical example of the differences between the two problems, observe that if the underlying graph is a disjoint union of paths, then solving Minimum Linear Arrangement optimally takes linear time, but even on these simple instances, Graphical House Allocation is NP-complete [20].

We do note, however, that all the hardness constructions by Hosseini et al. [20] used the disconnectedness of the underlying graphs crucially, in finding reductions from bin packing instances. Their results also show that for very simple classes of disconnected graphs, Graphical House Allocation is inapproximable to any finite factor. However, these proof techniques do not carry over to *connected* graphs, and so it was not known whether any of these reductions would go through for connected graphs. For instance, a well known result by Chung [7] states that Minimum Linear Arrangement is solvable in polynomial time on trees; the complexity of this problem for Graphical House Allocation was open.

1.1 Our Contributions

We present a complete characterization of the approximability of Graphical House Allocation on various classes of connected graphs, summarized in Table 1. In particular, for any instance on the following graph classes, we show a polynomial-time[1] algorithm on an n-vertex graph G (with maximum degree Δ) in that class for obtaining the stated multiplicative approximation to the optimal envy, and then demonstrate a matching lower bound that shows that any polynomial improvement on the approximation ratio is impossible on that graph class unless P = NP:

- If G is any connected graph, *any* allocation attains the trivial upper bound of $O(n^2)$ (Proposition 3.1). In Theorem 4.3, we show that we cannot have an $O(n^{2-\epsilon})$-approximation for any $\epsilon > 0$. We also give a polynomial-time $\widetilde{O}(\mathsf{treewidth}(G) \cdot \Delta)$-approximation algorithm (Corollary 3.5).
- If G is a tree, *any* allocation attains the trivial upper bound of $O(n)$ (Proposition 3.1). In Theorem 4.2, we show that we cannot have an $O(n^{1-\epsilon})$-approximation for any $\epsilon > 0$. This is in stark contrast to Minimum Linear Arrangement, where there are sub-quadratic algorithms for *exact* solutions on trees [7]. We also explicitly show a simple divide-and-conquer procedure (Algorithm 1) that gives the same $O(\Delta \log n)$-approximation in $O(n \log n)$ time.
- If G is planar, Corollary 3.5 gives us a polynomial-time algorithm to achieve an $\widetilde{O}(\sqrt{n\Delta})$-approximation to the optimal envy. In the worst case, $\Delta = \Theta(n)$, so this is a worst-case approximation of $\widetilde{O}(n)$. Once again, Theorem 4.2 shows that we cannot have an $O(n^{1-\epsilon})$-approximation for any $\epsilon > 0$.
- If G is a bounded-degree graph, Corollary 3.5 gives us a polynomial-time algorithm to achieve an $\widetilde{O}(\mathsf{treewidth}(G))$-approximation to the optimal envy. Again, this is a worst-case approximation of $\widetilde{O}(n)$. Using Theorem 4.6, we show that we cannot have an $O(n^{1-\epsilon})$-approximation for any $\epsilon > 0$. This is our most involved technical result, and it uses expansion properties of Ramanujan graphs.
- If G is a bounded-degree planar graph, Corollary 3.5 gives us a polynomial-time $\widetilde{O}(\sqrt{n})$-approximation algorithm. We match this by showing that we cannot have an $O(n^{0.5-\epsilon})$-approximation for any $\epsilon > 0$ (Theorem 4.5).
- If G is a bounded-degree tree, both Algorithm 1 and Corollary 3.5 give us a polynomial-time algorithm that outputs an $\widetilde{O}(1)$-approximation to the optimal envy. We show that finding the exact optimal envy is NP-hard (Theorem 4.10).

Note that assuming connectivity in the results above is necessary, since Hosseini et al. [20] showed that disconnected graphs cannot have the optimal envy approximated to any finite factor. We give the first known results for connected graphs.

We also show that for random graphs, any allocation is a $(1 + o(1))$-approximation with high probability (Theorem 3.7).

Finally, we investigate complete binary trees in further detail. We first show that the class of binary trees is not "well-behaved", by refuting a conjecture by Hosseini et al. [20] about the structural properties of exact optimal allocations on binary trees by means of a counterexample (Section 3.1). The hardness results in Theorems 4.2, 4.6, and 4.10 might have suggested that complete binary trees cannot have $o(\log n)$-approximations in general. We show, however, that just the in-order traversal on a complete binary tree achieves a 3.5-approximation to the optimal envy (Theorem 5.5). We also show that this approximation ratio cannot be improved beyond 1.67 by a natural class ("value-agnostic") of algorithms.

Our paper is organized as follows. In Section 2, we set up preliminaries. In Sections 3 and 4, we present our upper and lower bounds respectively from Table 1. In Section 5, we discuss binary trees. We finish with concluding remarks and open directions in Section 6. Due to space constraints, we present brief proof sketches of our main results; detailed proofs can be found in the full version [19].

1.2 Other Related Work

Our work is very close to the large body of results on the computability of Minimum Linear Arrangement. While finding optimal linear arrangements is intractable in general [12], there have been several papers presenting approximation algorithms for the problem [11, 13, 27], with the best known approximation ratio being $O(\sqrt{\log n} \log \log n)$ [13]. Note that it is relatively straightforward to show that an α-approximation algorithm for the Minimum Linear Arrangement problem yields an $\alpha\phi$ approximation for the Graphical House Allocation problem where $\phi = \max_{1 \le i \le n-1}(h_{i+1} - h_i)/\min_{1 \le i \le n-1}(h_{i+1} - h_i)$.

Our problem also generalizes the classical problem of Minimum Bisection, which asks how to partition a graph G into two almost equally-sized components with the smallest number of edges going across the cut. This problem is NP-complete [17] and it is also known to be inapproximable by an additive factor of $n^{2-\epsilon}$ [6]. These lower bounds carry over to the Graphical House Allocation

[1] In all our results, $\widetilde{O}$ hides $\mathrm{polylog}(n)$ factors

Approximations for Graphical House Allocation		
Graph Class	**Upper Bound**	**Lower Bound**
Connected graphs	$O(n^2)$ (Prop. 3.1) $O(\text{treewidth}(G) \cdot \Delta \log^{2.5} n)$ (Cor. 3.5(iii))	$\omega(n^{2-\epsilon})$ (Thm. 4.3)
Trees	$O(n)$ (Prop. 3.1) $O(\Delta \log n)$ (Alg. 1, Cor. 3.5(i))	$\omega(n^{1-\epsilon})$ (Thm. 4.2)
Planar graphs	$O(\sqrt{n\Delta} \log^{1.5} n)$ (Cor. 3.5(ii))	$\omega(n^{1-\epsilon})$ (Thm. 4.2)
Bounded-degree graphs	$O(\text{treewidth}(G) \cdot \log^{2.5} n)$ (Cor. 3.5(iii))	$\omega(n^{1-\epsilon})$ (Thm. 4.6)
Bounded-degree planar graphs	$O(\sqrt{n} \log^{1.5} n)$ (Cor. 3.5(ii))	$\omega(n^{0.5-\epsilon})$ (Thm. 4.5)
Bounded-degree trees	$O(\log n)$ (Thm. 3.3, Cor. 3.5(i))	> 1 (NP-hard, Thm. 4.10)
Random graphs	$1 + O(\sqrt{\ln(n)/n})$ (Thm. 3.7)) *w.h.p.*	–
Complete binary trees	3.5 (Thm. 5.5))	open (Conj. 6.1)

Table 1: Summary of our results. Here, Δ is the maximum degree of the graph in question, and the lower bounds assume P ≠ NP. Note that in all cases, the upper and lower bounds match up to polylogarithmic factors, showing that nontrivial improvements to these upper bounds are impossible unless P = NP. All our upper bounds are polynomial time.

problem as well, although the latter is strictly harder. For instance, Minimum Bisection is known to be solvable exactly in polynomial time for forests, but Graphical House Allocation is NP-hard [20].

The canonical house allocation problem has also been well-studied in the literature. Recall that, in the canonical house allocation problem, agents are allowed to disagree on the values of the houses. In this setting, the existence and computational complexity of envy-free allocations on graphs have been reasonably well-studied [3, 5, 10], with the problem, unsurprisingly, being computationally intractable in most settings. There has also been several lines of work studying the complexity of minimizing various notions of envy when the underlying graph is *complete* [1, 16, 21, 25]. For a detailed survey of various lines of work where there are graph-based constraints on the agents or goods, we refer the reader to Biswas et al. [4].

2 MODEL AND PRELIMINARIES

We have a set of n *agents* $V = [n]$ placed on the vertices of an undirected graph $G = (V, E)$. There are n *houses*, each with a non-negative *value*, that need to be allocated to the agents. We represent the houses simply by the multiset of values $H = \{h_1, \ldots, h_n\}$, and assume WLOG that $h_1 \leq \ldots \leq h_n$. We will interchangeably talk about the house with value h_i and the real number h_i. The pair (G, H) defines an *instance* of Graphical House Allocation.

An *allocation* $\pi : V \to H$ is a bijective mapping from agents (or nodes) to house values. Given an allocation π and an edge $(i, j) \in E$, we define the *envy* along the edge (i, j) as $|\pi(i) - \pi(j)|$. Our goal in Graphical House Allocation is to compute an allocation π^* that *minimizes* the total envy along all the edges of G:

$$\mathsf{Envy}(\pi, G) := \sum_{(i,j) \in E} |\pi(i) - \pi(j)|.$$

We adopt the following definition from Hosseini et al. [20] that provides a geometric representation to visualize allocations.

Definition 2.1 (Valuation Interval)**.** *For any instance (G, H) of Graphical House Allocation, define the* valuation interval *as the closed interval $[h_1, h_n] \subset \mathbb{R}_{\geq 0}$. For any allocation π, the envy along the edge $(i, j) \in E$ is exactly the length of the interval $[\pi(i), \pi(j)]$ (assuming $\pi(i) \leq \pi(j)$). We sometimes call the intervals $[h_i, h_{i+1}]$ for $1 \leq i \leq n-1$ the* smallest subintervals *of the valuation interval.*

An optimal allocation π^* would minimize the sum of the lengths of the intervals corresponding to each of its edges. An allocation π is *α-approximate* if $\mathsf{Envy}(\pi, G) \leq \alpha \cdot \mathsf{Envy}(\pi^*, G)$.

Fix any arbitrary class $\mathcal{G}$ of graphs (we allow $\mathcal{G}$ to be a singleton set). We say an algorithm $\mathsf{ALG}_{\mathcal{G}}$ is *defined* on $\mathcal{G}$ if $\mathsf{ALG}_{\mathcal{G}}$ is well-specified and outputs a valid allocation on every instance (G, H) of Graphical House Allocation with $G \in \mathcal{G}$. Such an algorithm $\mathsf{ALG}_{\mathcal{G}}$ is an *α-approximation* if for all instances (G, H) of Graphical House Allocation with $G \in \mathcal{G}$, $\mathsf{ALG}_{\mathcal{G}}$ always outputs an α-approximate allocation. A 1-approximation is an exact algorithm.

Definition 2.2 (Value-Agnostic Algorithms)**.** *An algorithm $\mathsf{ALG}_{\mathcal{G}}$ defined on a graph class $\mathcal{G}$ is* value-agnostic *if on every input (G, H) with $G \in \mathcal{G}$, $\mathsf{ALG}_{\mathcal{G}}$ returns the same allocation on all instances where the* ordering *of house values is the same (in other words, the algorithm only requires the ordinal ranking and not the numerical values). If the graph class $\mathcal{G}$ admits a value-agnostic α-approximation algorithm, we say $\mathcal{G}$ is* α-value-agnostic. *Otherwise, it is* α-value-sensitive.

How can we re-frame existing results on Graphical House Allocation in the light of Definition 2.2? Hosseini et al. [20] show that, unless P = NP, there is no 1-approximation algorithm $\mathsf{ALG}_{\mathcal{G}}$ when $\mathcal{G}$ is the set of vertex-disjoint unions of paths, cycles, or stars. In contrast, they show that value-agnostic *exact* algorithms exist when $\mathcal{G}$ is the set of paths, cycles, or stars.

Of course, value-agnostic α-approximations are extremely powerful algorithms, as they can exploit the graph structure *independently* of the values in the Graphical House Allocation instance. As we would expect, value-agnostic 1-approximations do not always exist, even on very simple graph classes and even if we allow

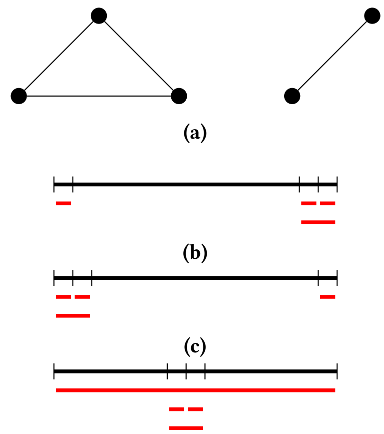

Figure 1: (a) shows the graph $K_2 \cup K_3$. (b), (c), and (d) show different valuation intervals with different optimal assignments for $K_2 \cup K_3$. In particular, note that every optimal solution for (b) requires the two smallest values to go to the K_2, but every optimal solution for (c) requires the three smallest values to go to the K_3, and an α-value-agnostic algorithm cannot distinguish between the two, for any finite α.

for unlimited time. For instance, consider the graph consisting of the disjoint union of K_2 and K_3. Figure 1 shows that this graph does not admit an α-value-agnostic algorithm for any finite α.

Although all our examples so far use the disconnectedness of the graphs to illustrate value-sensitivity, we will see in Section 5 that there are value-sensitive connected graphs as well.

For any graph $G = (V, E)$, and $S \subseteq V$, we denote by $\delta_G(S)$ the number of edges going across the cut $(S, V - S)$ in G. We will often estimate $\delta_G(S)$ for various subsets S. For $k \leq n - 1$, we define $\delta_G(k) := \min_{|S|=k} \delta_G(S)$ as the size of the smallest cut in G with k vertices on one side. Of course, $\delta_G(k) = \delta_G(n - k)$ for all k. A $(k, n - k)$-cut in G will be any cut $(S, V - S)$ with $|S| = k$.

We will use a few concepts from structural graph theory, most notably that of *cutwidth*.

Definition 2.3. *For a graph $G = (V, E)$ on n vertices, let $\sigma = (v_1, \ldots, v_n)$ be any ordering of V. The* width *of σ is defined as*

$$\mathsf{width}(\sigma, G) := \max_{1 \leq \ell \leq n-1} \delta_G(\{v_1, \ldots, v_\ell\}).$$

The cutwidth *of G is the minimum width over all orderings of G, i.e.,*

$$\mathsf{cutwidth}(G) := \min_{\sigma \in S_n} \mathsf{width}(\sigma, G) .$$

The ordering σ is often called a *layout*. An *optimal* layout is an ordering that achieves the cutwidth of G. The cutwidth is closely related to other standard notions of width used in structural graph theory. In particular, we have the following chain of inequalities (see Korach and Solel [23]):

$$\begin{aligned} &\mathsf{treewidth}(G) \leq \mathsf{pathwidth}(G) \leq \mathsf{cutwidth}(G) \\ &\quad \leq O(\Delta \cdot \mathsf{pathwidth}(G)) \leq O(\Delta \cdot \mathsf{treewidth}(G) \cdot \log n) \end{aligned} \quad (1)$$

Finding the exact cutwidth of G in general is a difficult algorithmic problem. It can be computed exactly for trees (along with an optimal ordering) in time $O(n \log n)$ [31]. However, even for planar graphs, the problem is NP-complete [26].

If G is sufficiently dense, there is a polynomial-time approximation scheme for the cutwidth [2]. In general, there is an efficient $O(\log^{1.5} n)$-approximation of the cutwidth known [24], which also returns a layout achieving this ratio. We will use this process as a subroutine several times in Section 3, for our upper bounds.

3 UPPER BOUNDS

The hardness of achieving optimal envy even on simple classes of graphs (e.g., disjoint unions of paths) [20] immediately gives rise to the question of whether we can approximate optimal solutions. As stated before, we need to assume connectivity in general.

We start by making a trivial observation (Proposition 3.1): *any* allocation of values to a connected graph is an $O(n^2)$-approximation to the optimal envy, and in fact an $O(n)$-approximation when the graph is a tree. This is due to the fact that every smallest subinterval of the valuation interval is covered by at most $|E|$ edges, but connectivity requires that it be covered by at least one edge.

Proposition 3.1. *For* any *instance of* Graphical House Allocation *on a connected graph $G = (V, E)$,* any *allocation is an $|E|$-approximation to the optimal value.*

In what follows, we first discuss how to improve this bound for bounded-degree trees and then generalize this result to graphs based on a structural parameter called the *cutwidth*. Finally, we showcase how our bounds can be significantly improved for the special class of random (Erdős-Renyi) graphs.

3.1 Trees

In this section, we present a recursive polynomial-time $(\Delta \log n)$-approximation algorithm for any instance of Graphical House Allocation where the underlying graph is any tree with maximum degree Δ. Thus, for any tree with maximum degree $\Delta = o(n/\log n)$, our algorithm provides a better approximation than Proposition 3.1.

We will use the following folklore fact[2] without a proof.

Fact 3.2 (Folklore). *Every n-vertex tree T has a* center of gravity*: i.e., a vertex v such that all connected components of $T - v$ have at most $n/2$ vertices. This vertex v can be found in $O(n)$ time.*

We will use Fact 3.2 in developing a recursive algorithm (Algorithm 1) that obtains an $O(\Delta \log n)$-approximation on trees. In each call, the algorithm first finds a center of gravity of the tree and subsequently uses this vertex to identify disjoint subtrees and solve the subproblems recursively on disjoint subintervals of the valuation interval.

Theorem 3.3. *There is an $O(n \log n)$-time algorithm that, given any instance on a tree with maximum degree Δ, returns an allocation whose envy is at most $\Delta \log n$ times the optimal envy.*

We remark that with a slightly more careful analysis,[3] we can improve the approximation ratio to $(1/2) \cdot (1 + \Delta + \Delta \log n)$. In particular, for any instance on a binary tree, the optimal envy can be $(2 \log n)$-approximated in $O(n \log n)$ time.

[2]For a proof of this fact, see, for instance, Chung [7], who attributes this as a folklore result to Seidvasser [28], who claims the fact is well-known, but proves it anyway.
[3]Technically this involves tweaking the algorithm such that the center of gravity is assigned slightly differently in line 7, and the partition of H is consistent with this.

Algorithm 1 Recursive Algorithm TrickleDown(T, H)

Input: A Graphical House Allocation instance on a tree T and a set of values $H = \{h_1, \dots, h_n\}$.
Output: An $O(\Delta \log n)$-approximate allocation.

```
if |T| = 1 then
    Allocate the only house to the only vertex.        ▷ Base case
else
    Find a center of gravity v of T.
    Let T − v = T_1 + ... + T_k, with |T_i| = n_i.     ▷ k ≤ Δ, n_i ≤ n/2.
    Partition H into the following contiguous sets:
        H_1 = {h_1, ..., h_{n_1}},
        H_2 = {h_{n_1+1}, ..., h_{n_1+n_2}}
        ⋮
        H_k = {h_{n_1+...+n_{k-1}+1}, ..., h_{n_1+...+n_k}}.
    Allocate h_n to vertex v.
    for i ∈ {1, ..., k} do
        Recursively call TrickleDown(T_i, H_i).
return the resulting allocation.
```

3.2 Cutwidth

In this section, we generalize the result from Section 3.1 using the structural graph theoretic property of cutwidth (Definition 2.3). This will enable us to have a black-box process to obtain envy approximations parameterized by the cutwidth. All of these algorithms will be value-agnostic.

Theorem 3.4. *Let (G, H) be a Graphical House Allocation instance defined on a connected graph G. Given a layout σ that β-approximates* $\mathrm{cutwidth}(G)$*, we can efficiently construct an allocation π that is a $(\beta \cdot \mathrm{cutwidth}(G))$-approximation to the optimal envy.*

The next corollary follows from Theorem 3.4 and Equation 1 when combined with existing bounds on the cutwidth, treewidth, or pathwidth [9, 23, 24] of certain graph families along with the best known approximation results of these quantities [24, 31].

Corollary 3.5. *There exist polynomial-time value-agnostic approximation algorithms for the following classes:*

(i) An $O(\Delta \log n)$-approximation algorithm on trees,
(ii) An $O(\sqrt{n\Delta} \log^{1.5} n)$-approximation algorithm on planar graphs,
(iii) An $O(\mathrm{treewidth}(G) \cdot \Delta \log^{2.5} n)$-approximation algorithm on general connected graphs.

Note that for each class of graphs listed above Δ can be $O(n)$ in the worst case, and for general connected graphs, $\mathrm{treewidth}(G)$ can be $O(n)$ in the worst case as well. So, in the worst case, the first and third results are asymptotically worse than the trivial bound given by Proposition 3.1. However, for many natural subclasses of these graphs, such as bounded-degree graphs and bounded-degree trees, Corollary 3.5 yields strictly better approximation guarantees.

3.3 Random Graphs

We next consider random graphs, specifically Erdős-Renyi graphs, where $G \sim \mathcal{G}_{n,1/2}$ denotes a random graph on n nodes where every edge is present with probability $1/2$ and all edges are independent. We show that Graphical House Allocation on such graphs can be approximated up to a factor $1 + o(1)$ regardless of the valuation interval. The central observation is that for any subset of nodes S, $\delta_G(S)$ is tightly concentrated around $|S|(n - |S|)/2$.

Lemma 3.6. *For $G \sim \mathcal{G}_{n,1/2}$,*

$$\Pr\left[\forall S \subseteq V, \left|1 - \frac{\delta_G(S)}{|S|(n-|S|)/2}\right| \le \epsilon\right] \ge 1 - \exp(-\Omega(\epsilon^2 n)),$$

for any $\epsilon \ge \sqrt{24 \ln(n)/n}$.

Lemma 3.6, with $\epsilon = \sqrt{24 \ln(n)/n}$, implies that with high probability, the cost of the optimum solution is at least

$$\sum_{i=1}^{n-1} (h_{i+1} - h_i)\delta_G(i) \ge \sum_{i=1}^{n-1} (h_{i+1} - h_i)(1-\epsilon)i(n-i)/2,$$

whereas the cost of an arbitrary allocation is at most

$$\sum_{i=1}^{n-1} (h_{i+1} - h_i)(1+\epsilon)i(n-i)/2.$$

Therefore, an arbitrary allocation has an approximation ratio of $(1+\epsilon)/(1-\epsilon) = 1 + O(\sqrt{\ln(n)/n})$.

Theorem 3.7. *For $G \sim \mathcal{G}_{n,1/2}$, any allocation is a $1+O(\sqrt{\ln(n)/n})$ approximation with probability at least $1 - 1/\mathrm{poly}(n)$.*

4 LOWER BOUNDS

Every algorithm presented in Section 3 is value-agnostic. It might seem reasonable to assume, therefore, that there are more powerful approximation schemes that exploit the numerical values in H in some way. Indeed, our results on random graphs suggest that, for most graphs, we can do significantly better. Remarkably, we show in this section that this is *not* the case, and our value-agnostic algorithms are strong enough to give us nearly optimal approximation guarantees. Specifically, we show inapproximability results matching our upper bounds (up to polylog factors) for every class of graphs considered. Our lower bounds will use reductions from the Unary 3-Partition problem.

Definition 4.1 (3-Partition). *Given a multiset of $3m$ naturals $A = \{a_1, \dots, a_{3m}\} \subseteq \mathbb{N}_{>0}$ and a natural $T \in \mathbb{N}_{>0}$ such that $\sum_{j \in [3m]} a_j = mT$, 3-Partition asks whether A can be partitioned into m triplets $(S_1, S_2, \dots, S_m)$ such that the sum of each triplet is equal to T.*

The 3-Partition problem is NP-complete even when all the inputs are given in unary and each item in A is strictly between $T/4$ and $T/2$ [18]. We refer to this variant as Unary 3-Partition. Note that Unary 3-Partition is just a reformulation of Bin Packing: there are $3m$ integers that sum to mT, and we wish to fit these integers into m bins each of capacity T. The condition of three integers in each bin is redundant, as it is implied by the constraint that each integer is strictly between $T/4$ and $T/2$.

Some of our results and proofs in this section (specifically Theorems 4.2 and 4.5) are very similar to results about the inapproximability of the *balanced graph partition* problem [14, 15]. The rest of our proofs use novel gadgets and techniques.

4.1 Trees and Planar Graphs

Recall that we presented two approximation guarantees for trees, $O(n)$ (Proposition 3.1) and $O(\Delta \log n)$ (Corollary 3.5). Both of these results are $\widetilde{O}(n)$ in the worst case.

Theorem 4.2. *For any constant $\epsilon > 0$, there is no efficient $O(n^{1-\epsilon})$ approximation algorithm for Graphical House Allocation on depth-2 trees unless P = NP.*

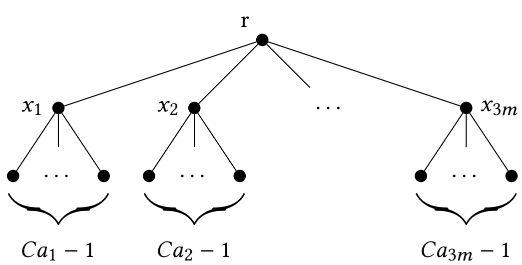

Figure 2: Mapping a Unary 3-Partition instance to a tree.

Proof Sketch. Given a Unary 3-Partition instance, we construct a graph according to Figure 2 where C is some positive integer we will decide later. The multiset of house values consists of CT houses with value j for each $j \in [m]$, and one house with value 0.

If there is a valid 3-partition, we can construct an allocation with envy at most $3m^2$. If there is no 3-partition, any allocation must have envy at least C. We can now set C appropriately. □

4.2 General and Bounded-Degree Graphs

In this section, we generalize the arguments from Section 4.1 to other classes of graphs. The main technique is similar to that of Theorem 4.2, so we just present ideas for the graph construction in each of these proofs, with the details in Hosseini et al. [19].

We first match the $O(n^2)$ upper bound for connected graphs (Proposition 3.1 and Corollary 3.5).

Theorem 4.3. *For any constant $\epsilon > 0$, there is no efficient $O(n^{2-\epsilon})$ approximation algorithm for Graphical House Allocation on connected graphs unless P = NP.*

Proof Sketch. We replace the Ca_i-sized stars in Figure 2 with Ca_i-sized cliques. The rest of the proof is similar to Theorem 4.2. □

So far in our two lower bounds (Theorems 4.2 and 4.3), we were able to use simple counting techniques, because counting edges with non-zero envy in stars and cliques is straightforward. Our next results will require much more careful analysis.

We will start with bounded-degree planar graphs. Our reduction uses grid graphs instead of stars and cliques, and so we will need a technical lemma to help us with estimating the number of edges with nonzero envy.

Lemma 4.4. *Let $G = Grid(r, c)$ be a grid graph with r rows and c columns such that $r \leq c$. Let $A \subseteq V$ be any set of nodes in this graph such that $|A| \leq rc/2$. Then, $\delta_G(A) \geq \min\{\sqrt{|A|}, r/2\}$.*

Armed with Lemma 4.4, we can now present our lower bound on bounded-degree planar graphs.

Theorem 4.5. *For any constant $\epsilon > 0$, no efficient $O(n^{0.5-\epsilon})$ approximation algorithm exists for Graphical House Allocation on bounded-degree planar graphs unless P = NP.*

Proof Sketch. We replace the stars of size Ca_i in Figure 2 with grid graphs containing C rows and Ca_i columns. The rest of the proof flows similarly to Theorem 4.2. Lemma 4.4 helps in estimating the envy blow-up if there is no 3-partition. □

Note that Theorem 4.5 matches the $O(\sqrt{n})$ upper bound from Corollary 3.5.

Our next lower bound applies to arbitrary graphs with bounded degree, and matches the $O(n)$ upper bound from Proposition 3.1 and Corollary 3.5. In this reduction, we use a recent polynomial-time algorithm [8] to compute bipartite Ramanujan multigraphs for any even number m of vertices, and any degree $d \geq 3$. At a high level, we replace the star gadgets from the proof of Theorem 4.2 with these Ramanujan graphs and use the expansion properties of Ramanujan graphs to prove a lemma similar to (and stronger than) Lemma 4.4.

Theorem 4.6. *For any constant $\epsilon > 0$, there is no efficient $O(n^{1-\epsilon})$ approximation algorithm for the Graphical House Allocation problem on bounded-degree graphs unless P = NP.*

4.3 Bounded-Degree Trees

Our final lower bound shows that Graphical House Allocation is NP-hard even when the underlying graph is a bounded degree tree. We still use Unary 3-Partition in our reduction but this proof is significantly different from the previous ones. Our reduction will use a gadget we call the *flower.*[4]

Definition 4.7. *The flower $F(n, k)$ is a rooted tree with n nodes and maximum degree $k + 1$, defined recursively as follows: for any $k \geq 1$, $F(1, k)$ is simply an isolated vertex which is the root node. For $n > 1$, $F(n, k)$ consists of a root node connected to the root nodes of d other flowers $F(n_1, k), \ldots, F(n_d, k)$ such that*

(a) $\sum_{i=1}^{d} n_i = n - 1$,
(b) if $n - 1 \geq k$, then $d = k$ if n and k have different parities, and $d = k - 1$ otherwise,
(c) each n_i is odd,
(d) for any $i, j \in [d]$, $|n_i - n_j| \leq 2$.

To ensure consistency with floral terminology, we refer to the root node of the flower $F(n, k)$ as its pistil *and the (recursively smaller) flowers $F(n_1, k), \ldots, F(n_d, k)$ as its* petals.

Before we use flowers, we show that they are well-defined and efficiently constructible.

Lemma 4.8. *For any $n \geq 1$ and $k \geq 3$, the flower $F(n, k)$ exists and can be constructed in poly(n, k) time.*

The reason we build flowers is because they satisfy the two following useful properties.

Lemma 4.9. *Let $F(n, k)$ be a flower on the set of vertices N, and suppose $n \geq 10k$, and n and k have different parities. Then, $F(n, k)$ satisfies the following properties:*

[4]To the best of our knowledge, our specific flower graph is novel but it is possible (likely even) that the term "flower" has appeared before in the graph theory literature.

(i) *For any $A \subseteq N$ such that $|A|$ is even and A does not contain the pistil, $\delta(A) \geq 2$.*
(ii) *Each petal of $F(n, k)$ has size in the interval $\left[\frac{4n}{5k}, \frac{6n}{5k}\right]$.*

These simple properties are all we need to show the hardness of Graphical House Allocation on bounded-degree trees.

Theorem 4.10. *Graphical House Allocation is NP-hard on bounded-degree trees.*

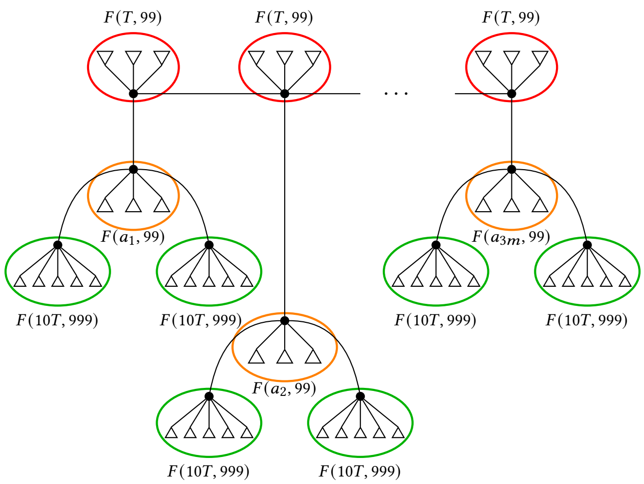

Figure 3: Mapping a Unary 3-Partition instance to a bounded degree tree. Here, the orange, red, and green circles correspond to small, medium, and large flowers respectively.

Proof Sketch. Given a Unary 3-Partition instance, we construct a graph according to Figure 3; the shaded circles correspond to pistils and the white triangular blocks correspond to petals. The house values are defined as follows: we have $4m + 1$ unique values such that the gaps between these values are exponentially decreasing. That is, the gap between the least and the second least value is significantly larger than the gap between the second least and the third least value and so on. For each unique value, there are T houses with that value in the multiset H, with the exception of the largest value which has enough houses (with that value) to ensure the total size of the multiset H is equal to the number of nodes in the graph.

We can show that in any optimal allocation, the first $3m$ clusters must be allocated to flowers of the form $F(T, 99)$. The next m clusters must be allocated in a way that creates a 3-partition to minimize envy. That is, each of these values must be allocated to three flowers of the form $F(a_i, 99)$ such that the total size of these three flowers sums up to T. If it is not possible to do this, the minimum envy of the allocation is strictly higher. This allows us to separate instances with a valid 3-Partition. □

5 THE CURIOUS CASE OF COMPLETE BINARY TREES

In this section, we investigate Graphical House Allocation on instances where the underlying graph is a complete binary tree B_k. Recall that such a tree has depth k, and $2^{k+1} - 1$ vertices in total, of which 2^k are leaves. All leaves, furthermore, are at the same depth.

In Hosseini et al. [20, Theorem 4.11], it was shown that for any binary tree (complete or otherwise), at least one optimal allocation satisfies the *local median property*: the value at every internal node is the median among the values given to that node and its two children. The same authors surmised that, for any binary tree, at least one optimal allocation satisfies the stronger *global median property*: for every internal node v, either its left subtree gets strictly lower-valued houses and its right subtree gets strictly higher-valued houses, or the other way round. Note that if true, this would lead to a straightforward recursive polynomial-time algorithm that would compute an optimal allocation on (nearly) balanced binary trees.

We now give a refutation of this conjecture. We illustrate an instance on a complete binary tree of depth 3, in which no optimal allocation satisfies the global median property. This is a quite surprising result that shows that the general problem on complete binary trees may be much harder than expected.

Example 5.1. *Consider the instance (B_3, H), where*

$$H = \{0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 3, 3, 3, 3\}.$$

See Figure 4. The top shows the only allocation satisfying the global median property (up to re-ordering). The total non-negligible envy incurred by this assignment comes out of the thick red edges of the B_3, which incur a total envy of 6. However, the bottom shows an allocation with an envy of 5 (incurred by the thick red edges), showing that the global median is strictly sub-optimal.

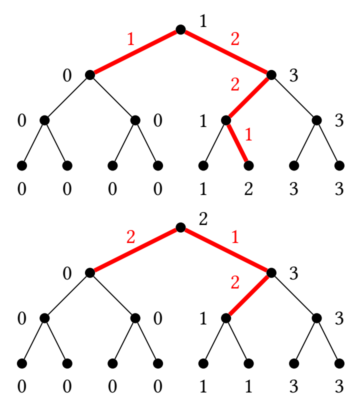

Figure 4: Refutation of the global median property on complete binary trees.

Fix an arbitrary instance of Graphical House Allocation on the complete binary tree B_k on $n = 2^{k+1} - 1$ vertices, and consider the valuation interval. There are n values on the interval. Of particular interest to us is the size of the *smallest* $(i, n-i)$-cut, i.e., $\delta_{B_k}(i)$. Since $\delta_{B_k}(i) = \delta_{B_k}(n-i)$, we can WLOG take $i \leq \lceil n/2 \rceil$. We now need a definition.

Definition 5.2 (Repunit Representation and Elegance)**.** *For any $m \geq 1$, let a* repunit representation *of m be any finite sequence*

m	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
elegance(m)	1	2	1	2	3	2	1	2	3	2	3	2	3	2	1	2	3	2	3	4

Table 2: List of elegance(m) **for** $1 \le m \le 20$**.**

$(a_1, \ldots, a_r) \in \mathbb{Z}^r$ satisfying

$$m = \sum_{i=1}^{r} \mathrm{sgn}(a_i) \cdot (2^{|a_i|} - 1)$$

where $\mathrm{sgn}(a_i)$ *is* 1 *(resp.* -1*) if $a_i \ge 0$ (resp. $a_i < 0$). Note that every $m \ge 1$ has a repunit representation (e.g., the length-m sequence of all ones). We define* elegance(m) *as the smallest r for which m has a repunit representation $(a_1, \ldots, a_r)$ of length r.*

The intuition behind this definition is to capture the most "efficient" way to write m as the sum or difference of binary repunits, i.e., numbers of the form $11\ldots1$. For instance, elegance$(10) = 2$, because $10 = (2^3 - 1) + (2^2 - 1)$, and there is no shorter repunit representation. Similarly, elegance$(12) = 2$, as $12 = (2^4 - 1) - (2^2 - 1)$. Note that 12 cannot be written as the *sum* of two repunits. Table 2 summarizes the elegance of all numbers up to 20.

The following proposition relates elegance to the size of the smallest $(i, n-i)$-cut in a complete binary tree, namely $\delta_{B_k}(i)$.

Proposition 5.3. *Let B_k be the complete binary tree on $n = 2^{k+1} - 1$ vertices. Then for $i \le 2^k - 1$,* elegance$(i) - 1 \le \delta_{B_k}(i) \le$ elegance(i).

Proof Sketch. For each edge going across a cut in B_k, one of its endpoints is the root of a binary subtree, and it contributes a term in a repunit representation (possibly along with an extra additive term). Conversely, any repunit representation gives rise to a cut. Therefore, cuts correspond to repunit representations up to a single additive term. Minimizing both sides yields the result. □

We note that if $i \ll n$, then in fact $\delta_{B_k}(i) =$ elegance(i). Therefore, elegance(i) actually characterizes the size of the minimum $(i, n-i)$ cut in any sufficiently large binary tree.

Consider a value-agnostic algorithm for complete binary trees. Such an algorithm would need to assign the house values in any instance in some fixed order $(v_1, \ldots, v_n)$ to the vertices of B_k. The following proposition shows that doing this cannot simultaneously achieve the optimal cut on all smallest subintervals, and this leads to a lower bound on the approximability.

Proposition 5.4. *There is no value-agnostic algorithm for complete binary trees that attains an approximation better than* $(5/3) \approx 1.67$.

The counterexample in Proposition 5.4 and the failure of the global median property (Example 5.1) may seem to suggest that, even for complete binary trees, *any* constant approximation ratio is unattainable. Remarkably, the following result shows that this is not the case: there is a *value-agnostic* algorithm attaining a constant approximation on any complete binary tree. Indeed, ordering the vertices of B_k in the standard in-order traversal and allocating the (sorted) values in that order yields a 3.5-approximation.

Theorem 5.5. *Let B_k be the complete binary tree on $n = 2^{k+1} - 1$ vertices. Then, on any house allocation instance on B_k, assigning the houses in increasing order to the vertices of B_k in the standard in-order traversal gives us a total envy at most* 3.5 *times the optimal value.*

It is instructive to check why this technique does not hold for arbitrary binary trees. Proposition 5.3 does not hold in general for non-complete binary trees. A complete binary tree ensures that there is always a binary subtree of the size given by a repunit representation to include on one side of the cut, but we lose this guarantee for non-complete trees.

We leave it as an open problem to construct either value-agnostic deterministic algorithms that achieve an approximation ratio better than 3.5, or to obtain any polynomial-time algorithm (which cannot be value-agnostic) to obtain any approximation ratio better than 1.67 for complete binary trees. We believe there should be an exact algorithm for this very special class of graphs, and hope that this will instigate future research into this problem.

6 CONCLUSIONS

We explored the approximability of Graphical House Allocation, presenting tight approximation algorithms for several classes of connected graphs, to our knowledge the first such results in the area. In particular, we gave polynomial-time algorithms exploiting graph structures to approximate the optimal envy on general graphs, trees, planar graphs, bounded-degree graphs, bounded-degree planar graphs, and bounded-degree trees; for each of these classes, we also gave a matching lower bound. Our algorithms were value-agnostic, i.e., they took into account only the input graph and the ordering among the house values but not the values themselves. We showed that any allocation on a random graph is a $(1 + o(1))$-approximation, and also gave a value-agnostic algorithm to show a 3.5-approximation on all instances on complete binary trees.

The main question we leave for future work is the complexity of Graphical House Allocation on complete binary trees. We know by the results in Section 5 that no exact algorithm can be value agnostic, but there seems to be no obvious way of leveraging the values, on even such a structured class of graphs.

Conjecture 6.1. *Graphical House Allocation is polynomial-time solvable on complete binary trees.*

ACKNOWLEDGMENTS

We wish to thank Justin Payan for many helpful discussions during the early stages of this work. We also wish to thank Paul Seymour for some discussions related to the material in Section 5. Rohit Vaish acknowledges support from SERB grant no. CRG/2022/002621 and DST INSPIRE grant no. DST/INSPIRE/04/2020/000107. Andrew McGregor and Rik Sengupta acknowledge support from NSF grant CCF-1934846. Hadi Hosseini acknowledges support from NSF IIS grants #2144413 and #2107173.

REFERENCES

[1] Elad Aigner-Horev and Erel Segal-Halevi. 2022. Envy-Free Matchings in Bipartite Graphs and their Applications to Fair Division. *Information Sciences* 587 (2022), 164–187.

[2] Sanjeev Arora, Alan Frieze, and Haim Kaplan. 1996. A New Rounding Procedure for the Assignment Problem with Applications to Dense Graph Arrangement Problems. In *Proceedings of 37th Conference on Foundations of Computer Science*. 21–30.

[3] Aurélie Beynier, Yann Chevaleyre, Laurent Gourvès, Julien Lesca, Nicolas Maudet, and Anaëlle Wilczynski. 2018. Local Envy-Freeness in House Allocation Problems. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. 292–300.

[4] Arpita Biswas, Justin Payan, Rik Sengupta, and Vignesh Viswanathan. 2023. *The Theory of Fair Allocation Under Structured Set Constraints*. 115–129. https://doi.org/10.1007/978-981-99-7184-8_7

[5] Robert Bredereck, Andrzej Kaczmarczyk, and Rolf Niedermeier. 2022. Envy-Free Allocations Respecting Social Networks. *Artificial Intelligence* 305 (2022), 103664.

[6] Thang Nguyen Bui and Curt Jones. 1992. Finding Good Approximate Vertex and Edge Partitions is NP-Hard. *Inform. Process. Lett.* 42, 3 (1992), 153–159.

[7] F.R.K. Chung. 1984. On Optimal Linear Arrangements of Trees. *Computers & Mathematics with Applications* 10, 1 (1984), 43–60.

[8] Michael B. Cohen. 2016. Ramanujan Graphs in Polynomial Time. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*. 276–281.

[9] Hristo N Djidjev and Imrich Vrt'o. 2006. Crossing Numbers and Cutwidths. *Journal of Graph Algorithms and Applications. v7* (2006), 245–251.

[10] Eduard Eiben, Robert Ganian, Thekla Hamm, and Sebastian Ordyniak. 2020. Parameterized Complexity of Envy-Free Resource Allocation in Social Networks. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*. 7135–7142.

[11] Guy Even, Joseph Seffi Naor, Satish Rao, and Baruch Schieber. 2000. Divide-and-Conquer Approximation Algorithms via Spreading Metrics. *J. ACM* 47, 4 (2000), 585–616.

[12] S. Even and Y. Shiloach. 1978. NP-Completeness of Several Arrangements Problems. *Technical Report, TR-43 The Technicon* (1978), 29.

[13] Uriel Feige and James R Lee. 2007. An Improved Approximation Ratio for the Minimum Linear Arrangement Problem. *Inform. Process. Lett.* 101, 1 (2007), 26–29.

[14] Andreas Emil Feldmann. 2012. Fast Balanced Partitioning Is Hard Even on Grids and Trees. In *Proceedings of the 37th International Symposium on Mathematical Foundations of Computer Science (MFCS)*. 372–382.

[15] Andreas Emil Feldmann and Luca Foschini. 2015. Balanced Partitions of Trees and Applications. *Algorithmica* 71, 2 (feb 2015), 354–376.

[16] Jiarui Gan, Warut Suksompong, and Alexandros A. Voudouris. 2019. Envy-Freeness in House Allocation Problems. *Mathematical Social Sciences* 101 (2019), 104–106.

[17] M.R. Garey, D.S. Johnson, and L. Stockmeyer. 1976. Some Simplified NP-Complete Graph Problems. *Theoretical Computer Science* 1, 3 (1976), 237–267.

[18] M. R. Garey and D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness* (first edition ed.). W. H. Freeman.

[19] Hadi Hosseini, Andrew McGregor, Rik Sengupta, Rohit Vaish, and Vignesh Viswanathan. 2023. Tight Approximations for Graphical House Allocation. arXiv:2307.12482 [cs.DS]

[20] Hadi Hosseini, Justin Payan, Rik Sengupta, Rohit Vaish, and Vignesh Viswanathan. 2023. Graphical House Allocation. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 161–169.

[21] Naoyuki Kamiyama. 2021. The Envy-Free Matching Problem with Pairwise Preferences. *Inform. Process. Lett.* 172 (2021), 106158.

[22] Naoyuki Kamiyama, Pasin Manurangsi, and Warut Suksompong. 2021. On the Complexity of Fair House Allocation. *Operations Research Letters* 49, 4 (2021), 572–577.

[23] Ephraim Korach and Nir Solel. 1993. Tree-Width, Path-Width, and Cutwidth. *Discrete Applied Mathematics* 43, 1 (1993), 97–101.

[24] Tom Leighton and Satish Rao. 1999. Multicommodity Max-Flow Min-Cut Theorems and Their Use in Designing Approximation Algorithms. *J. ACM* 46, 6 (1999), 787–832.

[25] Jayakrishnan Madathil, Neeldhara Misra, and Aditi Sethia. 2023. The Complexity of Minimizing Envy in House Allocation. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 2673–2675.

[26] Burkhard Monien and Ivan Hal Sudborough. 1988. Min Cut is NP-Complete for Edge Weighted Trees. *Theoretical Computer Science* 58, 1-3 (1988), 209–229.

[27] Satish Rao and Andréa W Richa. 2005. New Approximation Techniques for Some Linear Ordering Problems. *SIAM J. Comput.* 34, 2 (2005), 388–404.

[28] M. A. Seidvasser. 1970. The Optimal Number of Vertices of a Tree. *Diskref. Anal.* 19 (1970), 56–74.

[29] Lloyd Shapley and Herbert Scarf. 1974. On Cores and Indivisibility. *Journal of Mathematical Economics* 1, 1 (1974), 23–37.

[30] Lars-Gunnar Svensson. 1999. Strategy-Proof Allocation of Indivisible Goods. *Social Choice and Welfare* 16, 4 (1999), 557–567.

[31] Mihalis Yannakakis. 1985. A Polynomial Algorithm for the Min-Cut Linear Arrangement of Trees. *J. ACM* 32, 4 (1985), 950–988.