# Approximation of Hypergraph Coloring

Arash Rafiey*

**Abstract**

We study the approximability of Hypergraph Coloring, HC. We are given two hypergraphs $\mathcal{G}$ and $\mathcal{H}$ (assume the hyperedges are ordered) together with a cost function $c$, specifying the cost of coloring a given vertex of $\mathcal{G}$ with a given vertex of $\mathcal{H}$. The goal is to find a homomorphism, a.k.a. coloring, from $V(\mathcal{G})$ to $V(\mathcal{H})$ so that it preserves adjacency (the image of every hyperedge in $\mathcal{G}$ is a hyperedge in $\mathcal{H}$) and its cost (sum over individual cost) is minimized. When $\mathcal{H}$ is a fixed target hypergraph, we denote this problem by MHC($\mathcal{H}$). Some prominent problems that this framework captures are (Hypergraph) Vertex Cover, Min Sum $k$-Coloring, Multiway Cut, Min Ones, and others.

We show that if $\mathcal{H}$ is a $r$-uniform hypergraph with bounded width, then MHC($\mathcal{H}$) admits a constant approximation algorithm. We reduce the MHC($\mathcal{H}$) problem to a binary version that preserves the optimal cost. In the binary case, i.e., $\mathcal{H} = H$ is a digraph, we have a dichotomy: if $H$ has a bounded width, then MHC($H$) admits a constant factor approximation; otherwise, MHC($H$) does not admit any approximation.

Leveraging known inapproximability results for problems like the system of linear equations over $GF(2)$, such as 3LIN, we conjecture that if $r$-uniform hypergraph $\mathcal{H}$ does not have bounded width, then MHC($\mathcal{H}$) cannot be approximated within a constant factor unless **P**=**NP**.

Our work extends well beyond the scope of previous works, such as the Boolean case discussed by Khanna, Sudan, Trevisan, and Williamson [SICOMP 2001].

## 1 Introduction

Let $\mathcal{G}$ be a hypergraph. The vertex set of $\mathcal{G}$ is denoted by $V(\mathcal{G})$, and the hyperedges are denoted by $E(\mathcal{G})$. We assume that each hyperedge of $\mathcal{G}$ is an ordered tuple, meaning that $(x, y, z)$ and $(z, x, y)$ are distinct hyperedges. A hypergraph $\mathcal{G}$ is called $r$-uniform if all hyperedges have size $r$. A digraph is a 2-uniform hypergraph. A graph $H$ is a 2-uniform hypergraph when $(a, b)$ is a hyperedge of $H$, and $(b, a)$ is also a hyperedge of $H$, which is known as a symmetric digraph. This definition of graphs suits the purposes of this paper. However, it should be noted that symmetric digraphs differ from undirected graphs. For instance, all symmetric digraphs are Eulerian, whereas only some undirected graphs are.

To align with the standard terminology, we refer to the edges of a digraph as arcs, the edges of a graph as edges, and the edges of hypergraphs as hyperedges. When no ambiguity arises, we use the shorthand $u \in \mathcal{H}$ instead of $u \in V(\mathcal{H})$. When $\mathcal{H}$ is a digraph $H$, its arc set is denoted by $A(H)$, and we use the shorthand $uv \in A(H)$ instead of $(u, v) \in A(H)$. For a graph $H$, when two vertices $x$ and $y$ are adjacent, we denote the edge as $xy$ or $yx$.

*Computer Science, Indiana State University, Indiana, USA. Email: arash.rafiey@indstate.edu. Research supported in part by NSF grant 1751765.

A *homomorphism* of hypergraph $\mathcal{G}$ to a hypergraph $\mathcal{H}$, also known as an $\mathcal{H}$-Coloring of $\mathcal{G}$, is a mapping $f : V(\mathcal{G}) \to V(\mathcal{H})$, such that for each hyperedge $(x_1, x_2, \ldots, x_k) \in \mathcal{G}$, $(f(x_1), f(x_2), \ldots, f(x_k))$ is a hyperedge of $\mathcal{H}$. We say a mapping $f$ does not satisfy hyperedge $(x_1, x_2, \ldots, x_k)$, if $(f(x_1), f(x_2), \ldots, f(x_k))$ is not a hyperedge of $\mathcal{H}$. The homomorphism problem parameterized by target hypergraphs, denoted HC($\mathcal{H}$), takes a hypergraph $\mathcal{G}$ as input and asks whether there is a homomorphism from $\mathcal{G}$ to $\mathcal{H}$. Therefore, by fixing the hypergraph $\mathcal{H}$ we obtain a class of problems, one for each hypergraph $\mathcal{H}$.

For instance, HC($H$) when $H$ is an edge, is equivalent to the problem of determining whether the input graph $G$ is bipartite, known as the 2-Coloring problem. Similarly, if $H$ is a clique on $k$ vertices, then HC($H$) is the classical k-Coloring problem.

There are several optimization versions of HC($\mathcal{H}$) problem, two of which have attracted a lot of attention. One is to find a mapping $f : V(\mathcal{G}) \to V(\mathcal{H})$ that maximizes (minimizes) the number of satisfied (unsatisfied) hyperedges in $\mathcal{G}$. This problem is known under the name of Max CSP (Min CSP); an example is the Max Cut problem where the target graph $H$ is an edge. This line of research has received a lot of attention in the literature and there are very strong results concerning various aspects of approximability of Max CSP and Min CSP [2, 11, 15, 24, 28]. Note that the CSP's where *all* the hyperedges must be satisfied such as Vertex Cover and 3-Coloring problems are known as *strict* CSPs [27].

The focus of this paper is on an optimization version of the HC problem where we can express problems such as Vertex Cover and 3-Coloring. In this optimization version of HC($\mathcal{H}$) problem, we are not only interested in the existence of a homomorphism (i.e. satisfying *all* the hyperedges), but want to find the "best homomorphism". The Minimum Hypergraph Coloring problem to $\mathcal{H}$, denoted by MHC($\mathcal{H}$), for a given input hypergraph $\mathcal{G}$, and a cost function $c(x, i), x \in V(\mathcal{G}), i \in V(\mathcal{H})$, seeks a homomorphism $f$ of $\mathcal{G}$ to $\mathcal{H}$ that minimizes the total cost $\sum_{x \in V(\mathcal{G})} c(x, f(x))$. The cost function $c$ can take non-negative rational values.

| MHC($\mathcal{H}$): | |
|---|---|
| **Input:** | Hypergraph $\mathcal{G}$, and a cost function $c : V(\mathcal{G}) \times V(\mathcal{H}) \to \mathbb{Q}_{\geq 0}$. |
| **Objective:** | Find a *homomorphism* $f : V(\mathcal{G}) \to V(\mathcal{H})$ that minimizes $\sum_{x \in V(\mathcal{G})} c(x, f(x))$. |

The MHC problem offers a natural way to model and generalizes many optimization problems. One practical application of MHC (for bipartite graphs) in defense logistics is discussed in [13].

**Example 1.1** (Vertex Cover)**.** This problem can be seen as MHC($H$) where $V(H) = \{0, 1\}, E(H) = \{11, 01\}$, and $c(u, 0) = 0$, $c(u, 1) = 1$ for every $u \in V(G)$ where $G$ is the input graph.

For $k$-Hypergraph Vertex Cover, when the input is a hypergraph $\mathcal{G}$, the target hypergraph $\mathcal{H}$ consists of all the hyperedges $\{\{0, 1\}^t - (0, \ldots, 0), t \leq k\}$, and $c(u, 0) = 0$, $c(u, 1) = 1$ for every $u \in V(\mathcal{G})$ where $\mathcal{G}$ is the input hypergraph.

**Example 1.2** (Chromatic Sum)**.** In this problem, we are given a graph $G$, and the objective is to find a proper coloring of $G$ with colors $\{1, \ldots, k\}$ with *minimum color sum*. This can be seen as MHC($H$) where $H$ is a clique of size $k$ with $V(H) = \{1, \ldots, k\}$ and the cost function is defined as $c(u, i) = i$. The problem Chromatic Sum appears in many applications, such as resource allocation problems [3].

**Example 1.3** (Multiway Cut)**.** Let $G$ be a graph where each edge $e$ has a non-negative weight $w(e)$. There are also $k$ specific (terminal) vertices, $s_1, s_2, \ldots, s_k$ of $G$. The goal is to partition the vertices of $G$ into $k$ parts so that each part $i \in \{1, 2, \ldots, k\}$, contains $s_i$ and the sum of

the weights of the edges between different parts is minimized. Let $H$ be a graph with vertex set $\{a_1, a_2, \ldots, a_k\} \cup \{b_{i,j} \mid 1 \leq i < j \leq k\}$. The edge set of $H$ is $\{a_i a_i, a_i b_{i,j}, b_{i,j} a_j, a_j a_j \mid 1 \leq i < j \leq k\}$. Now obtain the graph $G'$ from $G$ by replacing every edge $e = uv$ of $G$ with the edges $ux_e, x_e v$ where $x_e$ is a new vertex. The cost function $c$ is as follows. $c(s_i, a_i) = 0$, else $c(s_i, d) = |G|$ for $d \neq a_i$. For every $u \in G \setminus \{s_1, s_2, \ldots, s_k\}$, set $c(u, s_i) = 0$. Set $c(x_e, b_{i,j}) = w(e)$. Now, finding a minimum multiway cut in $G$ is equivalent to solving MHC($H$) for $G'$ and $c$.

**Example 1.4** (Min-Ones for 3LIN)**.** We are given a set of equations of type $x_{i_1} \oplus x_{i_2} \oplus x_{i_3} = 0/1$. The goal is solve this system of equations so that the number of variables assigned to 1 is minimized. This is an instance of MHC($\mathcal{H}$) where $\mathcal{H} = \{(0,0,1), (0,1,0), (1,0,0), (1,1,1)\}$ and with the cost function $c(x_i, 0) = 0$, and $c(x_i, 1) = 1$.

**Example 1.5** (List Hypergraph Coloring (LHC))**.** LHC($\mathcal{H}$), seeks, for a given input hypergraph $\mathcal{G}$ and lists $L(x) \subseteq V(\mathcal{H}), x \in \mathcal{G}$, a homomorphism $f$ from $\mathcal{G}$ to $\mathcal{H}$ such that $f(x) \in L(x)$ for all $x \in \mathcal{G}$. This is equivalent to MHC($H$) (with total cost zero) with $c(u, i) = 0$ if $i \in L(u)$, otherwise, $c(u, i) = 1$. This problem is also known as List H-Coloring.

The MHC problem generalizes many other problems such as (Weighted) Min Ones [1, 9, 23], Min Sol [22], a large class of linear programs of bounded integers, Minimum Sum Coloring [3, 10, 26], and various optimum cost chromatic partition problems [14, 21, 25].

It is important to distinguish between MHC and Max CSPs. Unlike Max CSPs, where the objective is to maximize a payoff function associated with whether a constraint is satisfied or not, MHC require that all the hyperedges of the input to be mapped to hyperedges of the target. In other words, feasible solutions must satisfy all constraints, and solutions that satisfy only part of the constraints are not valid. One example of a Max CSP problem is Max Cut. The field of approximation for Max CSPs has been an active area of research for several decades, which was initiated in [15]. More recently, there has been a growing interest in approximating Max CSPs instances that are promised to be satisfiable [4, 5, 6, 7]. The approaches taken in these results have primarily relied on analytic methods, rather than leveraging the algebraic structure of the predicates.

In terms of graphs and digraphs, the complexity of *exact minimization* of MHC($H$) is well-understood. A complete complexity classifications were given in [12] for undirected graphs and in [20] for digraphs. More precisely, the result in [20] states that if $H$ has so-called *k-min-max* ordering, then MHC($H$) is polynomial time solvable and otherwise it is **NP**-complete.

There are only a few results on the approximability MHC parameterized by a target graph or digraph. The authors of [16] initiated the study of (constant factor) approximation algorithms of MHC($H$). They proved a dichotomy in the case of bipartite graphs. That is, for any fixed bipartite graph $H$, MHC($H$) is approximable within (constant) factor $|V(H)|$ if $H$ is a *co-circular arc* graph, otherwise MHC($H$) is not approximable unless $\mathbf{P} \neq \mathbf{NP}$. Interestingly, they showed such bipartite graphs can be characterized by the existence of a special type of vertex orderings. A bipartite graph is co-circular arc if and only if it admits a vertex ordering called *min ordering* [16]. This dichotomy result was extended to graphs in [29]. It was shown that for any fixed graph $H$, MHC($H$) is approximable within (constant) factor $|V(H)|$ if $H$ is a *bi-arc* graph, otherwise MHC($H$) is not approximable unless $\mathbf{P} \neq \mathbf{NP}$.

### 1.1 Our results and proof techniques:

For $r$-uniform hypergraph $\mathcal{H}$, we prove the following theorem.

**Theorem 1.6** (**Main theorem 1:**). *Let $\mathcal{H}$ be a $r$-uniform hypergraph. If $\mathcal{H}$ has a bounded width, i.e., width $(2,3)$, then MHC($\mathcal{H}$) admits a constant factor approximation.*

In terms of inapproximability, using Example 1.5 we observe the following.

**Observation 1.7.** Let $\mathcal{H}$ be a hypergraph. If LHC($\mathcal{H}$) is **NP**-complete then MHC($\mathcal{H}$) is not approximable within any factor, unless $\mathbf{P} = \mathbf{NP}$.

The dichotomy for the LHC problem [19] states that for digraph $H$, LHC($H$) is polynomial-time solvable if and only if $H$ does not contain a *digraph asteroidal triple* (DAT). The DAT-free digraphs are also known as *bounded width*, i.e. width $(2,3)$, digraphs (see [19]).

**Remark 1.8.** The concept of bounded width for hypergraphs differs from that for graphs and is unrelated to the size of the hyperedges or the number of hyperedges covering all the vertices of $\mathcal{H}$. Instead, it pertains to a hypergraph $\mathcal{H}$ for which LHC($H$) can be solved using the (2,3)-consistency check, which involves arc consistency and pair consistency (see Subsection 1.1.1 and Section 2 for further details).

Therefore, by Observation 1 and above discussion we have a stronger version of Theorem 1.6.

**Theorem 1.9** (**Main theorem 2: dichotomy for digraphs**). *Let $H$ be a digraph. If $H$ has a bounded width, i.e. width $(2,3)$, then MHC($H$) admits a constant factor approximation. Otherwise, MHC($H$) is not approximable unless **P=NP**.*

### 1.1.1 Algorithm and proof overview of Theorem 1.6

Let $\mathcal{G}$ together with cost function $c : V(\mathcal{G}) \times V(\mathcal{H}) \to \mathbb{Q}_{\geq 0} \cup \{+\infty\}$ be an instance of MHC($\mathcal{H}$) where $\mathcal{H}$ has bounded width. The goal is to find an assignment $f : V(\mathcal{G}) \to V(\mathcal{H})$ that is a homomorphism from $\mathcal{G}$ to $\mathcal{H}$, has cost less than $+\infty$, and minimizes $\sum_{x \in V(\mathcal{G})} c(x, f(x))$. We break down the proof of Theorem 1.6 into steps, and in each step, we highlight our techniques and their novelties.

**Step 1: Converting the instance into a binary case:** This is done by creating a digraph, denoted as $H = Bin(\mathcal{H})$, from $\mathcal{H}$, and another digraph, denoted as $D = Bin(\mathcal{G})$, from $\mathcal{G}$. The vertices of $H$ encompass those of $\mathcal{H}$, and for each hyperedge $\overline{\boldsymbol{\alpha}}$ of $\mathcal{H}$, a corresponding vertex $\boldsymbol{\alpha}$ is added in $H$. Subsequently, we add oriented paths, denoted as $P_{a,\boldsymbol{\alpha}}$, between any $a \in H \cap \mathcal{H}$ and any $\boldsymbol{\alpha} \in \mathcal{H}$ where hyperedge $\overline{\boldsymbol{\alpha}} \in \mathcal{G}$ contains $a$. Initially, the first arc of $P_{a,\boldsymbol{\alpha}}$ is set as forward $(* \to *)$. Beginning with the first coordinate in $\overline{\boldsymbol{\alpha}}$, at each step $j$, if $a$ appears in coordinate $j$ of $\overline{\boldsymbol{\alpha}}$, a forward arc is added to the end of the current vertex of $P_{a,\boldsymbol{\alpha}}$; otherwise, a zig-zag sequence of arcs (forward, backward, and then forward : $* \to * \leftarrow * \to *$) is added . Finally a forward arc is added from the current last vertex to $\boldsymbol{\alpha}$. So, the first and the last arcs in $P_{a,\boldsymbol{\alpha}}$ are forward arcs. Analogously, $D$ is constructed from $\mathcal{G}$. For every vertex $x \in D$ which is also a vertex of $\mathcal{G}$, and every vertex $a$ of $H$ which is also a vertex of $\mathcal{H}$, the cost is defined the same as $c(x, a)$. For every other vertex $y \in D$ and $a' \in H$, $c(y, a')$ is set to zero. This transformation is noteworthy because it does not change the value of the optimal homomorphism. More precisely, if there is a homomorphism $f$ from $\mathcal{G}$ to $\mathcal{H}$ with cost $W$ then there is a homomorphism $h$ from $D$ to $H$ with cost $W$. Conversely, if $h$ is a homomorphism from $D$ to $H$, then restriction of $h$ on vertices of $D$ corresponding the vertices of $\mathcal{G}$ is a homomorphism from $\mathcal{G}$ to $\mathcal{H}$ with the same cost.

**Step 2. Preprocessing and setting up the lists:** Recall that we are interested in finding a mapping from the vertices of $D$ to the vertices of $H$ that is a homomorphism, meaning that each arc of $D$ is mapped to an arc of $H$; thus, we prune the set of possible solutions by running a preprocessing procedure on the input instance. The main appeal of the preprocessing step is to prune the space of all possible mappings from $D$ to $H$. It starts by associating a list of possible images to each vertex $x$ of $D$, $L(x)$, initially $V(H) - \{a \in V(H) : c(x, a) = +\infty\}$, and to each pair of vertices $(x, y)$ of $V(D) \times V(D)$, a list $L^2(x, y)$, initially $L(x) \times L(y)$. Then the preprocessing gradually polishes and makes these lists smaller until no more changes are possible. Two powerful and well-studied preprocessing procedures are the so-called *arc consistency* and *pair consistency*, also known as $(2, 3)$-consistency. It is known that $(2, 3)$-consistency *solves* $\mathsf{LHC}(H)$ when $H$ has bounded width [19]. If after the $(2, 3)$-consistency procedure we end up with some empty lists, then there is no homomorphism from $D$ to $H$; otherwise, there is a homomorphism from $D$ to $H$. We assume the $(2, 3)$-consistency procedure is successfully executed on our instance.

**Step 3. LP formulation:** We formulate $\mathsf{MHC}(H)$ as an integer linear program. We introduce a simple yet powerful LP to facilitate our task regarding its rounding. We first discuss the issues of a simple LP that one might propose.

One simple LP formulation is the following. Define variables $x_{u,i} \in [0, 1]$ for every $u \in D$ and $i \in L(u)$. Now for every $u \in D$, we add the constraint $\sum_{i \in L(u)} x_{u,i} = 1$, and for every $u, v \in D$ we add the constraint $x_{u,i} \leq \sum_{(i,j) \in L^2(u,v)} x_{v,j}$. The objective function is minimizing $\sum_{u \in D} x_{u,i} c(u, i)$. However, we do not believe there is an easy way to round this LP and obtain a constant approximation factor.

We change the LP above by looking at the $\mathsf{MHC}(H)$ as a minimum cut problem and writing the LP constraints to address this issue. That is, we assume some ordering of the vertices of $H$, say $a_1, a_2, \ldots, a_p, a_{p+1}$, (where $a_{p+1}$ is just a new dummy vertex) and define variables $x_{u,a_i} \in [0, 1]$, $u \in D$ and $a_i \in H$. We add constraints $x_{u,a_i} \geq x_{u,a_{i+1}}$ with $x_{u,a_1} = 1$, and $x_{u,a_{p+1}} = 0$ for all $u \in D$. Moreover, for every $u, v \in D$, we add constraint $x_{u,a_i} - x_{u,a_{i+1}} \leq \sum_{(i,j) \in L^2(u,v)} x_{v,a_j} - x_{v,a_{j+1}}$. The objective function is minimizing $\sum_{u \in D} (x_{u,a_i} - x_{u,a_{i+1}}) c(u, a_i)$.

Since this ordering is arbitrary, this is still not clear how to find a reasonably simple rounding algorithm. Thus, our next technique is to find a special ordering to capture the structural properties of bounded width digraphs. In this regard, we turn $\mathsf{MHC}(H)$ to its bipartite version, namely $\mathsf{MHC}(B(H))$. Here $B(H)$ is a bipartite graph with vertex set $I$ (where $I$ is the vertex set of $H$) and $I' = \{a' \mid a \in I\}$; a copy of $I$. The edge set of $B(H)$ consists of $ab'$ ($a \in I$, $b' \in I'$) if and only if $ab$ is an arc of $H$. Likewise, we construct $B(D)$, and extend the lists by setting $L(u') = \{a' \mid a \in L(u)\}$. For every $u', v' \in B(D)$, let $L^2(u', v') = \{(a', b') \mid (a, b) \in L^2(u, v)\}$, and similarly for every $u, v' \in B(D)$, let $L^2(u, v') = \{(a, b') \mid (a, b) \in L^2(u, v)\}$. Finally, let $L^2(v', u) = \{(b', a) \mid (a, b') \in L^2(u, v')\}$. The cost function $c : V(D) \times V(H) \to Q_{\geq 0} \cup \{+\infty\}$ is extended to $\mathsf{MHC}(B(H))$ by setting $c(u', a') = c(u, a)$.

Let $a_1, a_2, \ldots, a_p$ be an ordering of the vertices in $I$, and $b_1, b_2, \ldots, b_p$ be an ordering of the vertices in $I'$ ($b_i$ is not necessarily the copy of $a_i$), so that $a_1, a_2, \ldots, a_p, b_1, b_2, \ldots, b_p$ is a min-max ordering. This means if $a_i b_j$ and $a_{i'} b_{j'}$ are edges of $B(H)$ with $i < i'$ and $j' < j$ then $a_i b_{j'}, a_{i'} b_j$ are also edges of $B(H)$. It is relatively straightforward to see that such an ordering exists.

The importance of the bipartization technique is to use the min-max ordering (sub-modular) property of $B(H)$ and guarantee a constant factor approximation for $\mathsf{MHC}(H)$.

We formulate a linear program $\mathcal{S}$ and show that its integral solutions correspond to the optimal

solutions of the $\mathsf{MHC}(H)$. Let $\pi$ be a permutation on $\{1, 2, \ldots, p\}$ where $b_{\pi(i)}$ is the copy of $a_i$. The variables in $\mathcal{S}$ are denoted as $x_{u,a_i}$ and $x_{u',b_{\pi(i)}}$, where $u, u' \in B(D)$ and $a_i, b_{\pi(i)} \in B(H)$. The constraints include $x_{u,a_i} \geq x_{u,a_{i+1}}$ and $x_{u',b_j} \geq x_{u',b_{j+1}}$, initialized with $x_{u,a_1} = x_{u',b_1} = 1$. Additionally, $x_{u,a_{p+1}} = x_{u',b_{p+1}} = 0$, where $a_{p+1}$ and $b_{p+1}$ represent two extra nodes introduced into $B(H)$. Furthermore, pair consistency constraints are imposed; for every $u, v \in D$, the constraint $x_{u,a_i} - x_{u,a_{i+1}} \leq \sum_{a_j:(a_i,a_j)\in L^2(u,v)} (x_{v,a_j} - x_{v,a_{j+1}})$ is added. The objective function is to minimize $2\sum_{u\in D} c(u, a_i)(x_{u,a_i} - x_{u,a_{i+1}})$.

Linear system $\mathcal{S}$ provides an integral solution due to min-max ordering, $a_1, a_2, \ldots, a_p, b_1, b_2, \ldots, b_p$, yielding a minimum homomorphism from $B(D)$ to $B(H)$. However, our interest lies in homomorphisms from $D$ to $H$. To accommodate this, we add additional constraints $x_{u,a_i} - x_{u,a_{i+1}} = x_{u',b_{\pi(i)}} - x_{u',b_{\pi(i)+1}}$ for every $u, u' \in B(D)$ and $a_i, b_{\pi(i)} \in B(H)$. These constraints enable us (in the integral solution for $\mathcal{S}$) to obtain a homomorphisms $f : V(B(D)) \to V(B(H))$ such that if $f(u) = a_i$, then $f(u') = b_{\pi(i)}$. The objective function remains $2\sum_{u\in D} c(u, a_i)(x_{u,a_i} - x_{u,a_{i+1}})$. (Refer to Table 1 for the LP formulation and additional details.) Incorporating the additional constraints cause LP $\mathcal{S}$ to loose its intergral solution in general.

**Step 3: Rounding techniques for the LP:** After solving system $\mathcal{S}$ and getting fractional values, we use a random variable $0 \leq X \leq 1$, sampled uniformly at random, to round the values of $\mathcal{S}$ and obtain a homomorphism $f : B(D) \to B(H)$. If $X \leq x_{u,a_i}$ then set $\chi_{u,a_i} = 1$, else set $\chi_{u,a_i} = 0$. Likewise, if $X \leq x_{v',b_j}$ then set $\chi_{v',b_j} = 1$, else set $\chi_{v',b_j} = 0$. Now for every $u \in B(D)$ set $f(u)$ to be $a_i$ if $\chi_{u,a_{i+1}} = 0$ and $\chi_{u,a_i} = 1$. Also for every $u' \in B(D)$ set $f(u') = b_j$ if $\chi_{u',b_j} = 1$ and $\chi_{v',b_{j+1}} = 0$.

**Step 3.2. Making $f$ a homomprhism from $D$ to $H$:** A significant challenge here is to adjust the homomorphism $f$ to ensure consistency on both sides. Developing an algorithm that achieves this goal constitutes a crucial contribution to our work in this section, which will find and inspire further applications. The algorithm is simple yet built on the structural properties of bounded width digraphs, which shows the beauty of our methods.

We apply the SHIFT procedure to ensure that $f$ is consistent on both sides of $B(D)$. Specifically, we adjust $f$ so that for every $u, u' \in B(D)$, if $f(u) = a_i$, then $f(u') = b_{\pi(i)}$. We start with a vertex $z$ where $f(z) = a_r$ and $f(z') = b_l$ with $l \neq \pi(r)$. Note that $z$ is chosen when $(a_r, a_{\pi^-(l)})$ is a special pair (discussed later). We set $f(z') = b_{\pi(r)}$ (or $f(z) = a_{\pi^{-1}(l)}$).

The algorithm then initiates a special breadth-first search (SBFS) in $D \times H$ to update the images of all in-neighbors $u$ of $z$ due to the change in $f(z')$. A vertex $a_t$ with $(a_t, a_r) \in L^2(u, z)$ is selected if

$$\sum_{l=1}^{t} P_{u,a_l} < Y_0 \leq \sum_{l=1}^{t+1} P_{u,a_l} \tag{1.1}$$

where $P_{u,a_l} = (x_{u,a_l} - x_{u,a_{l+1}})/P_u$ and $P_u = \sum_{(a_l,a_r)\in L^2(u,z)} (x_{u,a_l} - x_{u,a_{l+1}})$.

We set $f(u) = a_t$ and also set $f(u') = b_{\pi(t)}$. The SHIFT procedure continues modifying the images of other vertices, such as $v$, which is adjacent to $u$ (either as an in-neighbor or out-neighbor) due to the change in $f(u)$ (or $f(u')$ respectively). A neighbor of $a_t$, say $a_s \in L(v)$, where $(a_s, a_t) \in L^2(v, u)$, is chosen with probability proportional to $x_{v,a_s} - x_{v,a_{s+1}}$ (according to $Y_1$ similar to the above equation). Essentially, we proceed with an SBFS in $D \times H$, adjusting $f$ as needed. When we next

need to adjust the image of $w$, a neighbor of $v$, we follow the same procedure, using the random variable $Y_0$. If further changes are required, we use $Y_1$, and so forth.

We leverage the fact that the random variables $X$, $Y_0$, and $Y_1$ are independent. Moreover, we show that the probability of shifting the image of $u$ to $a_t$ is bounded by $x_{u,a_t} - x_{u,a_{t+1}}$. Thus, we can use the random variable $Y_0$ to shift the image of $w$ (a neighbor of $v$) to some $a_l$ with a probability bounded by $x_{w,a_l} - x_{w,a_{l+1}}$. This enables us to use the random variables $Y_0$ and $Y_1$ periodically.

**Techniques for the correctness proof:** The correctness proof is given in Lemma 3.3 and Lemma 3.5. The approximation ratio is shown in Lemma 3.14.

The novelty of our approach is the careful exploiting of property of bounded width hypergraph $\mathcal{H}$ by introducing an auxilary digraph $H^{+2}$ (which has bounded width) and analyzing its strong components. Again here is another place that we show structural analysis matters when it comes to designing approximation algorithm for MHC.

The vertex set of $H^{+2}$ is $\{(a,b) \mid a,b \in \mathcal{H} \quad and \quad a \neq b\}$, and the arc set of $H^{+2}$ consists of $(a,b)(a',b')$ where there is an oriented path $P$ from $a$ to $a'$ going through exactly one vertex $\boldsymbol{\alpha} \in H$ (corresponding to a hyperedge of $\mathcal{H}$) and an oriented path $Q$ from $b$ to $b'$ (going through exactly one $\boldsymbol{\beta} \in H$ corresponding to a hyperedge of $\mathcal{H}$) so that $P$ and $Q$ are congruent (with the same length and follow the same patterns of forward and backward arcs) but there is no walk $R$ from $a$ to $b'$ and congruent with $P, Q$.

The vertices of $H^{+2}$ can be partitioned into two sets $A_1$ and $A_2$ where there exists a conservative polymorphism $\phi$ of $\mathcal{H}$ where its restriction on the vertices in $A_1$ is semilattice, and a conservative polymorphism $\psi$ of $\mathcal{H}$ where its restriction on the vertices in $A_2$ is majority [19]. $A_2$ would be the set of vertices containing both $(a,b)$ and $(b,a)$. There are several interesting properties for the strong components of $H^{+2}$; helping us in the correctness of the algorithm.

The procedure Shift goes through some vertices of $\mathcal{G} \times H^{+2}$ (the product digraph of $\mathcal{G}$ and $H^{+2}$) which are essentially based on the strong components of $H^{+2}$. When Shift reaches a sink component, it stays on that component until no further changes of images are necessary. This requires showing that Shift does not encounter a *circuit*. A circuit is formed if the procedure Shift changes the image of some vertex $x$ from $b_0$ to $b_1$, and then from $b_1$ to $b_2$, and eventually from some $b_k$ to $b_0$ again. It turns out that all the pairs $(b_1,b_0),(b_2,b_1),\ldots,(b_k,b_{k-1}),(b_0,b_k)$ must be in the same strong component $S$ of $H^{+2}$. Note that since $b_i \in L(x)$, there is a path from $b_i$ to $b_i$ in the list of the vertices of any oriented cycle containing $x$. We show that the Shift would leave the component $S$ and move to another component. When we reach a sink component, then no circuit can occur, and the procedure of shifting the images stops. This means at some point image of vertex $x \in D$, no longer changes to some vertex $b \in L(x)$. Therefore, Shift stops and returns a homomorphism from $D$ to $H$.

**Remark 1.10.** It is worth noting that the class of DAT-free digraphs (bounded width) is more general than the classes of digraphs that admit min ordering (known as bi-arc digraphs) and those that admit majority operations. Our LP formulation is simpler compared to those used in previous works such as [16, 29]. Proving the correctness of our approach for bounded width digraphs necessitates several new concepts and leverages the structural properties of these digraphs.

## 1.2 Future directions

**On the side of the upper bound.** We have shown MHC is approximable within a constant factor when $\mathcal{H}$ has bounded width. The constant factor of our approximation algorithm depends on the size of $\mathcal{H}$ which is assumed to be fixed. One natural question is for which $r$-uniform hypergraph $\mathcal{H}$, the approximation factor for MHC($\mathcal{H}$) is independent of the size of $\mathcal{H}$.

**On the inapproximability and lower bound** One apparent direction is to settle down the following conjecture:

**Conjecture 1.11.** Let $H$ be a $r$-uniform hypergraph. If $\mathcal{H}$ has not bounded width, then MHC($\mathcal{H}$) is not constant factor approximable unless **P=NP**.

The appeal for the above conjecture is the following: MHC($\mathcal{H}$) might admit an approximation algorithm only if LHC($\mathcal{H}$) is polynomial-time solvable. LHC($\mathcal{H}$) can be solved in polynomial time if $\mathcal{H}$ can be partitioned into a "bounded width" part and an "affine" part. Otherwise, this LHC is **NP**-complete [8]. For the affine case, it was observed in [23] that, based on the hardness result for Nearest Codeword problem [1], Min-Ones for 3LIN is not possible to approximate within a factor of $\Omega(2^{\log^{1-\varepsilon} n})$, unless **NP** $\subseteq$ **QP**.

Assuming Conjecture 1.11 holds, we can derive the following statement using Theorem 1.6:

**Statement 1.12.** *Let $\mathcal{H}$ be a $r$-uniform hypergraph. If $\mathcal{H}$ has bounded width then MHC($\mathcal{H}$) admits a constant factor approximation. Otherwise, MHC($\mathcal{H}$) is not constant factor approximable unless **P=NP**.*

Another important questions is the following. Let $\mathcal{H}$ be a hypergraph and not necessarily a $r$-uniform ones. *What is a dichotomy classification for MHC($\mathcal{H}$)?*

To answer this question one class of hypergraphs to consider is the class of bounded width hypergraphs. For the bounded width case, Lemma 8.14 of [23] shows that Min Ones for Horn SAT cannot be approximated to within a factor of $\Omega(2^{\log^{1-\varepsilon} n})$, unless **NP** $\subseteq$ **QP**. From these two hardness results, it was also noted by authors in [23] that Min Horn Deletion problem does not admit a constant approximation algorithm. Using their example, we give an example of hypergraph $\mathcal{H}$ consisting of three uniform hypergraphs (different sizes) for which MHC($\mathcal{H}$) does not admit a constant approximation.

The Min Horn Deletion problem consists of clauses in which at most one literal appears to be positive. The goal is to find an assignment to minimize the number of variables assigned to true. This is equivalent to consider clauses of form $(x \vee \neg y \vee \neg z) \wedge (\neg u \vee \neg v) \wedge (w \vee \neg p) \wedge (\neg p)$. If we translate this to a homomorphism problem, then we will have relations $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3$ of arity $3, 2, 2$ respectively from the set $\{x_1, x_2, \ldots, x_n\}$ and target relations $\mathbb{H}_1 = \{(0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,1), (1,1,0), (1,1,1)\}$, $\mathbb{H}_2 = \{(0,1), (1,0), (1,1)\}$, and $\mathbb{H}_3 = \{(0,0), (0,1), (1,1)\}$. The goal is to find an assignment for the variable so that every tuple in $\mathbb{G}_i$ is mapped to its corresponding $\mathbb{H}_i$, $i = 1, 2, 3$.

Now, define hypergraph $\mathcal{H}$ consists of hypergraphs $\mathcal{H}_1 = \{(0,0,0,a), (0,0,1,a), (0,1,0,a), (0,1,1,a), (1,0,1,a), (1,1,0,a), (1,1,1,a)\}$, $\mathcal{H}_2 = \{(0,1), (1,0), (1,1)\}$, and $\mathcal{H}_3 = \{(0,0,a), (0,1,a), (1,1,a)\}$. Moreover, define input hypergraph $\mathcal{G}$ as follows. For each tuple $(x,y,z) \in \mathbb{G}_1$ add $(x,y,z,\omega)$ into $\mathcal{G}$, for every $(x,y) \in \mathbb{G}_2$ add $(x,y,\omega)$ to $\mathcal{G}$, and for every $(x,y) \in \mathbb{G}_3$ add $(x,y,\omega)$ to $\mathcal{G}_3$. Define the cost function $c(\omega, a) = 0$, and $c(x,0) = 0$ and $c(x,1) = 1$ for every $x \neq \omega$. Then MHC($\mathcal{H}$) is equivalent to Min Horn Deletion problem, and hence, it does not admit a constant approximation algorithm.

The above discussion suggest, the constant factor approximable cases of $\mathsf{MHC}(\mathcal{H})$ should be bounded width and have at most two *uniform* sub-hypergraphs.

## 2 Definitions and preliminaries

**Polymorphisms and vertex orderings.** Let $\mathcal{R}$ be a $k$-uniform hypergraph on a set $\mathcal{A}$ and $\psi$ an $n$-ary operation on the same set. Operation $\psi$ is said to be a *polymorphism* of $\mathcal{R}$ if for any $\mathbf{a}^1, \ldots, \mathbf{a}^n \in \mathcal{R}$ the hyperedge $\psi(\mathbf{a}^1, \ldots, \mathbf{a}^n)$ belongs to $\mathcal{R}$. Here by $\psi(\mathbf{a}^1, \ldots, \mathbf{a}^n)$ we understand the component-wise action of $\psi$, that is, if $\mathbf{a}^i = (a^i_1, \ldots, a^i_k)$ then

$$\psi(\mathbf{a}^1, \ldots, \mathbf{a}^n) = (\psi(a^1_1, \ldots, a^n_1), \ldots, \psi(a^1_k, \ldots, a^n_k)).$$

Specifically a polymorphism of digraph $H$ of arity $k$ is a mapping $\phi$ from the set of $k$-tuples over $V(H)$ to $V(H)$ such that if $x_iy_i \in A(H)$ for $i = 1, 2, \ldots, k$, then $\phi(x_1, x_2, \ldots, x_k)\phi(y_1, y_2, \ldots, y_k) \in A(H)$. If $\phi$ is a polymorphism of $H$ we also say that $H$ admits the polymorphism $\phi$. A polymorphism $\phi$ is *idempotent* if it satisfies $\phi(x, x, \ldots, x) = x$ for all $x \in V(H)$, and is *conservative* if $\phi(x_1, x_2, \ldots, x_k) \in \{x_1, x_2, \ldots, x_k\}$. A binary polymorphism $\phi$ is called *commutative* if $\phi(a, b) = \phi(b, a)$ for all $a, b$, and is called *associative* if $\phi(\phi(a, b), c) = \phi(a, \phi(b, c))$ for all $a, b, c$. A commutative and associative polymorphism is called a *semilattice*. A ternary polymorphism $\psi$ is called *majority* if $\psi(x, x, y) = \psi(x, y, x) = \psi(y, x, x) = x$ for all $x, y$. In this paper, conservative semilattice polymorphisms (which are binary by definition) and conservative majority polymorphisms (which are ternary by definition) are of special interest.

A conservative semilattice polymorphism $\phi$ of $H$ naturally defines a binary relation $x \leq y$ on the vertices of $H$ by $x \leq y$ if and only if $\phi(x, y) = x$; by associativity, the relation $\leq$ is a linear order on $V(H)$, which we call the *min ordering* of $H$ associated with $\phi$.

**Definition 2.1.** An ordering $v_1 < v_2 < \cdots < v_n$ of $V(H)$ is a

- *min ordering* if and only if $uv, u'v' \in A(H)$ with $u < u'$ and $v' < v$ imply that $uv' \in A(H)$;
- *min-max ordering* if and only if $uv, u'v' \in A(H)$ with $u < u'$ and $v' < v$ imply that $uv', u'v \in A(H)$.

For a given bipartite graph $H = (B, W)$, let $\overrightarrow{H}$ be the digraph obtained by orienting all the edges of $H$ from $B$ to $W$. One can easily see that if $\overrightarrow{H}$ admits a min ordering then there is an ordering $a_1 < a_2 < \cdots < a_p$ of the vertices in $B$ and an ordering $b_1 < b_2 < \cdots < b_q$ of the vertices in $W$ so that $a_1 < a_2 < \cdots < a_p < b_1 < b_2 < \cdots < b_q$ is a min ordering of $\overrightarrow{H}$, in other words, $\overrightarrow{H}$ has a min ordering where vertices in $B$ precede vertices in $W$. We call such an ordering a *min ordering* for the bipartite graph $H$. Alternatively, we say bipartite graph $H = (B, W)$ admits a min ordering, if there is an ordering $a_1 < a_2 < \cdots < a_p$ of the vertices in $B$ and an ordering $b_1 < b_2 < \cdots < b_q$ of the vertices in $W$ so that if $a_ib_j$ and $a_{i'}b_{j'}$ with $i < i'$ and $j' < j$ are edges of $H$, then $a_ib_{j'}$ is also an edge of $H$. Similar treatment is applied for defining min-max ordering for bipartite graph $H$.

**Assumption 2.2.** In what follows when we mention walk, path, and cycle we mean oriented walk, oriented path, and oriented cycle, respectively, unless specified otherwise.

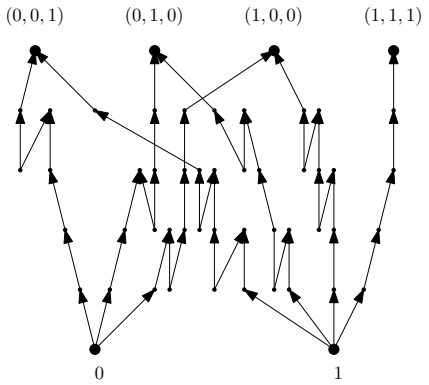


Figure 1: Conversion of $\mathcal{H} = \{(0,0,1),(0,1,0),(1,0,0),(1,1,1)\}$ to a digraph

**(Oriented) path, cycle and avoidance definition.** Let $H$ be a digraph. We say that $uv \in A(H)$ is an arc from $u$ to $v$. Sometimes, we emphasize this by saying that $uv$ is a *forward* arc of $H$, and also say $vu$ is a *backward* arc of $H$. For a walk $P = x_0, x_1, \ldots, x_n$ and any $i \leq j$, we denote by $P[x_i, x_j]$ the walk $x_i, x_{i+1}, \ldots, x_j$. We call $P[x_i, x_j]$ a *prefix* of $P$ if $i = 0$. For two walks, $P$ and $Q$ where the end vertex of $P$ is the same as the beginning of $Q$, let $PQ$ be the walk obtained from concatenation of $P$ and $Q$. We define two walks $P = x_0, x_1, \ldots, x_n$ and $Q = y_0, y_1, \ldots, y_n$ in $H$ to be *congruent* if they follow the same pattern of forward and backward arcs, i.e., $x_i x_{i+1}$ is a forward arc if and only if $y_i y_{i+1}$ is a forward arc, and $x_i x_{i+1}$ is a backward arc if and only if $y_i y_{i+1}$ is a backward arc. Suppose that the walks $P, Q$ as above are congruent. We say an arc $x_i y_{i+1}$ is *a faithful arc from $P$ to $Q$* if it is a forward (backward) arc when $x_i x_{i+1}$ is a forward (backward) arc (respectively). A *faithful arc from $Q$ to $P$* is defined symmetrically. We say $P$ *avoids* $Q$ if there is no faithful arc from $P$ to $Q$. We say $P$ and $Q$ *avoid each other* if $P$ avoids $Q$ and $Q$ avoids $P$. Notice that if $P$ avoids $Q$ then $Q^{-1}$ avoids $P^{-1}$ ($P^{-1}$ is the reverse of $P$).

**Definition 2.3** (Construction of digraph from a hypergraph $Bin(\mathcal{H})$)**.** Let $\mathcal{H}$ be a hypergraph. Let $Bin(\mathcal{H})$ be a digraph $H$ whose vertex set contains of a copy of $V(\mathcal{H})$ together with set $\{\boldsymbol{\alpha} \mid \overline{\boldsymbol{\alpha}} = (d_1, d_2, \ldots, d_k) \in E(\mathcal{H})\}$. Moreover, for each $d \in \mathcal{H}$ and each hyperedge $\overline{\boldsymbol{\alpha}}$ we add a new (oriented) path $P_{d,\boldsymbol{\alpha}}$ between $d \in V(H)$ and $\boldsymbol{\alpha}$ into $H$ if $d$ appears in $\overline{\boldsymbol{\alpha}} = (d_1, d_2, \ldots, d_r)$ as follows. The first arc of $P_{d,\boldsymbol{\alpha}}$ is a forward arc $dd^1$. At each step $1 \leq i \leq k$, if $d$ appears in the coordinate $i$ of $(d_1, d_2, \ldots, d_r)$, then add arc $d^i d^{i+1}$, otherwise, add arcs $d^i e^{i+1}, l^i e^{i+1}, l^i d^{i+1}$ (we call $e^{i+1}, l^i$ internal vertices). Finally add arc $d^k \boldsymbol{\alpha}$ (see figure 1). For every vertex $\tau \in H$, let $Base(\tau) = d$ when $\tau \in P_{d,\boldsymbol{\alpha}}$. Notice that $Base(\tau)$ is unique vertex of $H$ which is a copy of a vertex in $V(\mathcal{H})$. We say path $P$ in $H$ from $a \in H \cap \mathcal{H}$ to $b \in H \cap \mathcal{H}$ is minimal if it goes through exactly one vertex $\boldsymbol{\alpha} \in H$ corresponding to a hyperedge of $\mathcal{H}$.

Recall the definition of $H^{+2}$ in the previous section constructed from $\mathcal{H}$. In general we will also define $H^{+k}$ constructed from $\mathcal{H}$. We only need this definition for $k = 3$, to explain the obstruction for majority/semilattice polymorphism for $\mathcal{H}$.

**Definition 2.4** ($H^{+k} = Bin(\mathcal{H})^{+k}$)**.** Let $\mathcal{H}$ be a hypegraph. Let $H = Bin(\mathcal{H})$. Define $H^{+k} = Bin(\mathcal{H})^{+k}$ to be the digraph with the vertex set $\{(a_1, a_2, \ldots, a_k) | a_i \in V(\mathcal{H}), 1 \leq i \leq k\}$ and consisting of arcs $(a_1, a_2, \ldots, a_k)(b_1, b_2, \ldots, b_k)$ so that :

- for $1 \leq i \leq k$, there is minimal path $P_{a_i,b_i}$ from $a_i$ to $b_i$ in $H$ (see definition 2.3).
- All the $P_{a_i,b_i}$'s are congruent.
- For every $2 \leq j \leq k$, there is no path $P$ from $a_1$ to $b_j$ and congruent with $P_{a_1,b_1}$.

When $k = 2$, we say $(x, y)$, with $x \neq y$, is an *invertible pair* of hypergraph $\mathcal{H}$ if $(x, y)$ and $(y, x)$ belong to the same strong component of $H^{+2}$.

**Remark 2.5.** Let $\phi$ be a polymorphism on $\mathcal{H}$. Then, it is easy to extend $\phi$ on $H$ so that the properties of $\phi$ on the vertices of $H$ that are copies of the vertices in $\mathcal{H}$ are preserved.

**Definition 2.6** ($u \rightsquigarrow v$)**.** For two vertices $u, v$ of digraph $G$, let $u \rightsquigarrow v$ denote that $v$ is reachable from $u$ via a directed path in $G$. We write $x \not\rightsquigarrow y$, if there is no directed path from $x$ to $y$ in $G$. When we write $P : u \rightsquigarrow v$, we mean $P$ is a directed path from $u$ to $v$ in $G$.

**Definition 2.7** (DAT)**.** A digraph asteroidal triple (DAT) of $\mathcal{H}$ is an induced sub-hypergraph of $\mathcal{H}$ that yields three directed paths in $P_1, P_2, P_3$ in $H^{+3} = Bin(\mathcal{H})^{+3}$ where $P_1 : (a, b, c) \rightsquigarrow (p, q, q)$, $P_2 : (b, a, c) \rightsquigarrow (p, q, q)$, and $P_3 : (c, a, b) \rightsquigarrow (p, q, q)$ where $(p, q)$ is an invertible pair. (see [19]).

If $\mathcal{H}$ contains a DAT (as described above), then it is not difficult to see that all three pairs $(a, b), (b, c), (c, a)$ are invertible. Note that an invertible pair obstructs the existence of a conservative semilattice polymorphism. To see this, if $(a, b)$ is invertible in $\mathcal{H}$, then every pair $(a', b')$ in the same strong component as $(a, b)$ is also invertible. We show there is no conservative semilattice polymorphism $\phi$ on $H$. For contradiction, suppose $f$ is such a polymorphism. By the definition of $H^{+2}$, when $(a, b)(a', b')$ is an arc of $H^{+2}$ then $\phi(a, b) = a$ implies that $\phi(a', b') = a'$. Since $(a, b)$ is an invertible pair, there is a directed path $P$ from $(a, b)$ to $(b, a)$ in $H^{+2}$, and there exists a directed path $Q$ from $(b, a)$ to $(a, b)$ in $H^{+2}$. Now $\phi(a, b) \neq a$, as otherwise, by following $P$, we conclude that $\phi(b, a) = b$ which contradicts that $f$ is commutative. Similarly, $\phi(b, a) \neq b$, as otherwise, by following $Q$ we get a contradiction, and hence, $\phi$ is not semilattice. This would imply that if $(a, b)$ is invertible then $\phi$ is not semilattice on $a, b$.

Moreover, $H$ does not admit a conservative majority polymorphism $\psi$; because of $P_1$, $\psi(a, b, c) \neq a$, because of $P_2$ we have $\psi(a, b, c) \neq b$, and finally, because of $P_3$ we have $\psi(a, b, c) \neq c$. Therefore, the value of $\psi(a, b, c)$ can not be any of the $a, b, c$. A *permutable triple* in hypergraph $\mathcal{H}$ consists of three distinct vertices $a, b, c$, and three directed path $P_1, P_2, P_3$ in $H^{+3}$ where $P_1$ is from $(a, b, c)$ to a triple $(p_a, q_b, q_b)$, $P_2$ is from $(b, c, a)$ to a triple $(p_b, q_c, q_c)$ and $P_3$ is from $(c, a, b)$ to a triple $(p_c, q_a, q_a)$. Notice that the argument presented above implies that if $H$ contains a permutable triple, then it does not admit a conservative majority polymorphism. Indeed, the converse is also true.

**Theorem 2.8** ([19])**.** *Let $H$ be a digraph. Then the following hold.*

1. *$H$ has bounded width (DAT-free) if and only if there exists a conservative binary polymorphism $\phi$ and a conservative ternary polymorphism $\psi$ of $H$ such that for every $a, b \in V(H)$, either $\phi|_{\{a,b\}}$ is a semilattice polymorphism or $\psi|_{\{a,b\}}$ is a majority polymorphism. Furthermore, if $(a, b)$ is an invertible pair then $\phi(a, b) = a$ and $\phi(b, a) = b$.*
2. *LHC($H$) is polynomial-time if $H$ has bounded width, otherwise, LHC($H$) is NP-complete.*

Note that $\phi|_{\{a,b\}}$ is the restriction of $\phi$ over $a, b$ and likewise, $\psi|_{\{a,b\}}$ is the restriction of $\psi$ over $a, b$. That is $\psi$ on $\{a, b\}$ is majority.

**Definition 2.9.** Let $\mathcal{H}$ be a $r$-uniform hypergraph. We say $\mathcal{H}$ has bounded width if there exists a conservative binary polymorphism $\phi$ and a conservative ternary polymorphism $\psi$ of $H$ such that for every $a, b \in V(H)$, either $\phi|_{\{a,b\}}$ is a semilattice polymorphism or $\psi|_{\{a,b\}}$ is a majority polymorphism. Furthermore, if $(a, b)$ is an invertible pair then $\phi(a, b) = a$ and $\phi(b, a) = b$. Alternatively, $\mathsf{LHC}(\mathcal{H})$ can be solved by applying (2,3)-consistency.

**Definition 2.10** (G bipartization)**.** Let $G$ be a digraph with vertex set $I$. Let $B(G) = (I, I', E)$ be a bipartite graph with partite sets $I$ and $I'$ where $I'$ is a copy of $I$ and $ij'$ ($i \in I$, $j' \in I'$) is an edge of $B(G)$ if and only if $ij$ is an arc of $G$.

# 3 The approximation algorithm

This section is dedicated to our approximation algorithm. The description of the algorithm is given in Section 3.4. We prove that our approximation algorithm is correct in Section 3.5, and obtain an upper bound on the approximation ratio in Section 3.6.

## 3.1 Transformation:

Let $\mathcal{H}$ be the target $r$-uniform hypergraph and $\mathcal{G}$ be the input hypergraph together with the cost function $c : V(\mathcal{G}) \times V(\mathcal{H}) \to \mathbb{Q}_{\geq 0} \cup \{+\infty\}$.

**Construction of digraphs $H = Bin(\mathcal{H})$ and $D = Bin(\mathcal{G})$.** Let $H = Bin(\mathcal{H})$ and $D = Bin(\mathcal{G})$.

The cost function for vertices of $H$ that appear in $\mathcal{H}$ is the same. For every other vertex $y \in D$ and $a' \in H$, $c(y, a') = 0$.

**Lemma 3.1.** *Let $f$ be a homomorphism from $\mathcal{G}$ to $\mathcal{H}$ with the minimum cost $W$. Then there is a homomorphism $h$ from $D$ to $H$ with cost $W$. Conversely, if $h$ is a homomorphism from $D$ to $H$, then restriction of $h$ on vertices of $D$ corresponding the vertices of $\mathcal{G}$ is a homomorphism from $\mathcal{G}$ to $\mathcal{H}$ with the same cost.*

*Proof.* Let $f : V(\mathcal{G}) \to V(\mathcal{H})$. For each vertex $\mathbf{x} \in D$, where $\overline{\mathbf{x}} = (x_1, x_2, \ldots, x_r) \in \mathcal{G}$, define $h(\mathbf{x}) = \mathbf{a}$, where $\overline{\mathbf{a}} = (f(x_1), f(x_2), \ldots, f(x_r)) \in \mathcal{H}$. Extending $h$ is straightforward; it maps path $P_{y_1,\mathbf{y}}$ in $D$ to the path $P_{a,\boldsymbol{\alpha}}$ in $H$, where $f(y_1) = a$ and $f(\mathbf{y}) = \boldsymbol{\alpha}$. Since, $P_{y_1,\mathbf{y}}$ and $P_{a,\boldsymbol{\alpha}}$ are congruent, the intermediate vertices of $P_{y_1,\mathbf{y}}$ are mapped to their corresponding vertices in $P_{a,\boldsymbol{\alpha}}$. Based on the construction of $D$ and $H$ from $\mathcal{G}$ and $\mathcal{H}$, along with the provided cost function, the cost of $h$ is $W$.

Conversely, let $f$ represents a homomorphism from $D$ to $H$. Then, the restriction of $f$ to vertices of $D$ that are copies of vertices of $\mathcal{G}$ forms a homomorphism from $\mathcal{G}$ to $\mathcal{H}$. □

## 3.2 Preprocessing and consistency checks

Before presenting the system of linear equations $\mathcal{S}$ formulating the $\mathsf{MHC}$ problem, we give a procedure to associate lists to the vertices of $D$ and, subsequently, modify them based on some consistency conditions.

**Introducing lists.** To each vertex $x \in D$, we associate a list $L(x) \subseteq V(H)$ that is initially set to $V(H) - \{a \in V(H) : c(x,a) = +\infty\}$ — think of $L(x)$ as the "current" set of possible images for $x$ in a homomorphism from $D$ to $H$. We also consider a pair list $L^2(x,y)$ for every pair $x \neq y \in D$ — think of $L^2(x,y)$ as the ordered pairs of "current" possible images of $x, y$ in any of the homomorphisms from $D$ to $H$.

**Preprocessing to update lists and pair lists [(2,3)-consistency].** This procedure begins by performing arc consistency (2-consistency) and pair consistency (3-consistency) checks on lists ($L$ lists) and pair lists ($L^2$ lists) for digraph $D$. Arc consistency and pair consistency checks are standard procedures in graph/digraph homomorphism problems [17].

The arc consistency check is performed as follows. If $xy$ is an arc of $D$ and there exists $a \in L(x)$ such that $a$ does not have any out-neighbor in $L(y)$, then we remove $a$ from $L(x)$. Similarly, if there exists $b \in L(y)$ such that $b$ does not have any in-neighbor in $L(x)$, then we remove $b$ from $L(y)$. We continue this process until no list can be modified.

After the arc consistency check, we perform the pair consistency check. After the arc consistency process, the $L^2$ lists are initialized by setting $L^2(x,y) = \{(a,b) \mid a \in L(x), b \in L(y)\}$ for every $x, y \in D$. Now for every $x, y \in D$ and every $(a,b) \in L^2(x,y)$, if there exists $z$ such that for every $c \in L(z)$ either $(a,c) \notin L^2(x,z)$ or $(b,c) \notin L^2(y,z)$ then remove $(a,b)$ from $L^2(x,y)$. We continue this process until no list can be modified. If for some $a \in L(x)$, there is some $y \in D$ so that $a$ does not appear as the first coordinate of any pair in $L^2(x,y)$, then $a$ is removed from $L(x)$. In the end, if any list is empty, then clearly, there is no homomorphism from $D$ to $H$. Therefore, in the rest of the paper, we assume that all lists are non-empty.

**Fixing an ordering for $H$ using $B(H)$.** Let $a_1, a_2, \ldots, a_p$ be the vertices of $H$, where if $a_i b_r, a_i b_s, a_j b_r$ are edges of $B(H)$ then $a_j < a_i$ and $b_r < b_s$. Notice that since there are no induced paths of length 4 or more in $B(H)$, we can easily find such an ordering and notice that this ordering is a min-max ordering for $B(H)$. Let $\pi$ be a permutation on $\{1, 2, \ldots, p\}$ where $b_{\pi(i)}$ is the corresponding copy of $a_i$, $1 \leq i \leq p$.

**Extending lists $L$ and $L^2$ to vertices of $B(D)$ and $B(D) \times B(D)$** We introduce the lists $L$ for vertices in $B(D)$. Recall that the vertices of $B(D)$ are $v$ and $v'$ where $v'$ is the copy of $v \in D$. For every vertex $v' \in B(D)$ which is the copy of $v \in D$, set $L(v') = \{b_i \mid a_{\pi^{-1}(i)} \in L(v)\}$, and the list for the vertex $v \in B(D)$ is the same as the list in $D$. For every $u, v \in B(D)$, let $L^2(u,v) = \{(a_i, a_j) \mid (a_i, a_j) \in L^2(u,v), u, v \in D\}$. For every $u', v' \in B(D)$, let $L^2(u', v') = \{(b_i, b_j) \mid (a_{\pi^{-1}(i)}, a_{\pi^{-1}(j)}) \in L^2(u,v)\}$ and similarly for every $u, v' \in B(D)$, let $L^2(u, v') = \{(a_i, b_j) \mid (a_i, a_{\pi^{-1}(j)}) \in L^2(u,v)\}$. Finally, let $L^2(v', u) = \{(b_j, a_i) \mid (a_i, b_j) \in L^2(u, v')\}$.

### 3.3 Linear Program (LP) formulation

Now, we start to formulate a linear system $\mathcal{S}$ that will be shown to be equivalent to the $\mathsf{MHC}(\mathcal{H})$ and $\mathsf{MHC}(H)$ problem (Lemma 3.3). For all vertices $v, v'$ of $B(D)$ and $a_i, b_{\pi(i)}$ of $B(H)$ introduce the variables $x_{v,a_i}$ and $x_{v',b_{\pi(i)}}$. We also set $c(v', b_{\pi(i)}) = c(v, a_i)$.

Constraint (C6) is added because we need a special type of homomorphism $f$ from $B(D)$ to $B(H)$; the ones satisfying $f(u) = a_i$ and $f(u') = b_{\pi(i)}$ for all $u, u' \in B(D)$ and $a_i \in B(H)$. Notice that, by the definition of $L(v')$, $v' \in B(D)$, and constraint $(C7)$ we have :

Minimize: $\sum_{v\in B(D),a_i\in B(H)} c(v,a_i)(x_{v,a_i}-x_{v,a_{i+1}}) + \sum_{v'\in B(D),b_j\in B(H)} c(v',b_j)(x_{v',b_j}-x_{v',b_{j+1}})$

Subject to:

(C1) $x_{v,a_i}, x_{v',b_{\pi(i)}} \geq 0 \quad \forall v \in B(D),\ a_i \in B(H)$

(C2) $x_{v,a_1} = x_{v',b_1} = 1 \quad \forall v \in B(D)$

(C3) $x_{v,a_{p+1}} = x_{v',b_{p+1}} = 0 \quad \forall v, v' \in B(D)$ here $a_{p+1}, b_{p+1}$ are dummy vertices

(C4) $x_{v,a_{i+1}} \leq x_{v,a_i}$ and $x_{v',b_{\pi(i)+1}} \leq x_{v',b_{\pi(i)}} \quad \forall v \in B(D),\ a_i \in B(H)$

(C5) $x_{v,a_{i+1}} = x_{v,a_i}$ and $x_{v',b_{\pi(i)+1}} = x_{v',b_{\pi(i)}} \quad \forall v \in B(D), a_i \in B(H)$ if $a_i \notin L(v)$

(C6) $x_{u,a_i} - x_{u,a_{i+1}} = x_{u',b_{\pi(i)}} - x_{u',b_{\pi(i)+1}} \quad \forall u \in B(D),\ a_i \in B(H)$

(C7) $x_{u,a_i} - x_{u,a_{i+1}} \leq \sum_{a_j:(a_i,a_j)\in L^2(u,v)} (x_{v,a_j} - x_{v,a_{j+1}}) \quad \forall u, v \text{ in } B(D),\ a_i \in B(H)$

Table 1: LP relaxation $\mathcal{S}$

(C7-1) $x_{u,a_i} - x_{u,a_{i+1}} \leq \sum_{b_j:(a_i,b_j)\in L^2(u,v')} (x_{v',b_j} - x_{v',b_{j+1}}) \quad \forall u, v' \text{ in } B(D),\ a_i \in B(H)$ .

(C7-2) $x_{v',b_j} - x_{v',b_{j+1}} \leq \sum_{a_i:(a_i,b_j)\in L^2(u,v')} (x_{u,a_i} - x_{u,a_{i+1}}) \quad \forall u, v' \text{ in } B(D),\ b_j \in B(H)$ .

(C7-3) $x_{u',b_i} - x_{u',b_{i+1}} \leq \sum_{b_j:(b_i,b_j)\in L^2(u',v')} (x_{v',b_j} - x_{v',b_{j+1}}) \quad \forall u', v' \text{ in } B(D),\ b_i \in B(H)$.

The following lemma follows immediately from the construction of $B(H)$ and $B(D)$.

**Lemma 3.2.** *There exists a homomorphism $g : V(D) \to V(H)$ with cost $W$ if and only if there exists a homomorphism $f : V(B(D)) \to V(B(H))$ with cost $2W$ such that if $f(v) = a_i$ then $f(v') = b_{\pi(i)}$.*

### 3.4 LP rounding and finding a homomorphism from D to H

We start with an overview of our algorithm. The correctness and approximation bound proofs are postponed for the later subsections.

By Lemma 3.3, the integral solutions of $\mathcal{S}$ are in one-to-one correspondence to the homomorphisms from $D$ to $H$, and by by Lemma 3.1, they correspond to a homomorphism from $\mathcal{G}$ to $\mathcal{H}$ with the minimum total cost. Our algorithm will minimize the cost function over $\mathcal{S}$ in polynomial time using a linear programming algorithm. This will generally result in a solution with fractional values. We will obtain an integral solution by a randomized procedure called RoundingD-Shifting.

**Using random variable $X$ to get a partial homomorphism.** We select a random variable $X$ from the interval $[0, 1]$ uniformly, and then define the rounded values $\chi_{u,a_i} = 1$ if $X \leq x_{u,a_i}$; otherwise, we set $\chi_{u,a_i} = 0$. Similarly, for $\chi_{v',b_j}$, we set it to 1 if $X \leq x_{v',b_j}$, otherwise, it is set to 0.

We establish the *partial* homomorphism $f : V(B(D)) \to V(B(H))$ by assigning $f(u) = a_i$ when $\chi_{u,a_i} = 1$, and $\chi_{u,a_{i+1}} = 0$ (notably, $a_i$ is unique for each $u \in B(D)$). Likewise, for $f(v') = b_j$, we set it to $b_j$ if $\chi_{v',b_j} = 1$ and $\chi_{v',b_{j+1}} = 0$. It is easy to see that $f$ indeed is a homorphism from $B(D)$ to $B(H)$ but it may not be a consistent homomorphism; meaning that for some $u$, $f(u) = a_i$ and $f(u') \neq b_{\pi(i)}$. Let $U = V(D)$, and $U'$ be the copy of the vertices of $U$ in $B(D)$.

**Shifting to make $f$ consistent on both $U$ and $U'$ and obtaining a homomorphism from $D$ to $H$.**

We say a vertex $z$ of $B(D)$ is *unstable* if $f(z) = a_r$ and $f(z') \neq b_{\pi(r)}$, i.e., if $\chi_{z,a_r} = 1, \chi_{z,b_{\pi(r)}} = 0$, in which case one has $\chi_{z',b_l} = 1$, $\chi_{z',b_{l+1}} = 0$ for some $l \neq \pi(r)$. The goal of this step is to modify $f$ so that it remains a homomorphsim from $B(D)$ to $B(H)$ and also becomes consistent on both $U$ and $U'$, in the sense that there will be no unstable vertex $z \in U$. Note that $a_r \in L(z)$ if and only if $b_{\pi(r)} \in L(z')$. The algorithm starts with a special breadth first search (SBFS) in $B(D) \times B(H)$ and continues as long as there exists some unstable vertex in $B(D)$. Let $\phi$ be a semilattice on some pairs of $\mathcal{H}$ according to definition 2.9. The SBFS uses the following rules.

1. The SBFS starts with some $a_r \in V(H)$ (say with maximum index) for which there is vertex $z$, where $f(z) = a_r$ and $z$ is unstable. Suppose $f(z') = b_j$ where $j \neq r$. Let $a_{r'} = Base(a_r)$ and $a_{j'} = Base(a_j)$ (see definition 3.1 for constructing $H$ and $D$ ).

   If $\phi(a_{r'}, a_{j'}) = a_{r'}$ then SBFS sets $f(z') = b_{\pi(r)}$ otherwise it sets $f(z) = a_j$.

2. In general, when SBFS visits a pair $(u, a_i)$ then it shifts the image of $u$ to $a_i$ and sets $f(u)$ to $a_i$ and it sets $f(u') = b_{\pi(i)}$.

3. SBFS moves from $(u, a_i)$ with $u \in D$ to pair $(v, a_j)$ with $uv \in A(D)$ ($vu \in A(D)$) and $a_i f(v) \notin A(H)$, where $(a_i, a_j) \in L^2(u, v)$. Notice that the set of such $a_j$ is not empty due to arc consistency check. Among all candidates, $a_j$ is chosen at random with probability proportional to $x_{v',b_j} - x_{v',b_{j+1}} = x_{v,a_{\pi^{-1}(j)}} - x_{v,a_{\pi^{-1}(j+1)}}$, guided by random variable $Y_0$ or $Y_1$ depending on the parity (see lines 14,15 of SHIFT procedure). If the image of $u$ was shifted according to random variable $Y_l$, $l = 0, 1$ then the image of $v$ is shifted according to random variable $Y_{l+1}$ (sum is module 2)

4. When $(u, a_i)$ is visited, then it is added to the queue to start from $(u, a_i)$ at some later point.

**Algorithm 1** Rounding the fractional values of $\mathcal{S}$

1: **procedure** ROUNDING-SHIFTING($\mathcal{S}$)
2: Let $\{x_{u,a_i}\}$ and $\{x_{u',b_{\pi(i)}}\}$ be the (fractional) values returned by solving $\mathcal{S}$
3: Sample $X \in [0, 1]$ uniformly at random
4: For all $x_{u,a_i}$ : if $X \leq x_{u,a_i}$ set $\chi_{u,a_i} = 1$, else set $\chi_{u,a_i} = 0$
5: For all $x_{v',b_i}$ : if $X \leq x_{v',b_i}$ set $\chi_{v,b_i} = 1$, otherwise set $\chi_{v,b_i} = 0$
6: Set $f(u) = a_i$ where $\chi_{u,a_i} = 1$, $\chi_{u,a_{i+1}} = 0$
7: Set $f(v') = b_j$ where $\chi_{v',b_j} = 1$, $\chi_{v',b_{j+1}} = 0$
▷ At this point $f$ is a homomorphism from $B(D)$ to $B(H)$.
8: SHIFT($f$)
9: **return** $f$ ▷ $f$ is a homomorphism from $D$ to $H$.

**Algorithm 2** Procedure SHIFT

**procedure** SHIFT($f$)
  Let Q be an empty Queue, and let $Y_0, Y_1$ be two independent random variable from $[0,1]$.
  Let $\phi$ be the semilattice on $\mathcal{H}$, defined in Theorem 2.8
  **while** $\exists a_r \in H$ so that for some $z \in D$, $f(z) = a_r$ and $z$ is unstable **do** ▷ Maximum index $r$
    Let $a_r = f(z)$, and $b_{\pi(j)} = f(z')$ where $j \neq r$
    Let $a_{r'} = Base(a_r)$, and $a_{j'} = Base(a_j)$.
    **if** $\phi(a_{r'}, a_{j'}) = a_{r'}$ **then** set $f(z') = b_{\pi(r)}$ and $Q.enqueue(z, a_r, Y_0)$
    **else** set $f(z) = a_j$ and $Q.enqueue(z, a_j, Y_0)$
    **while** Q is not empty **do**
      $(u, a_i, Y_\lambda) \leftarrow Q.dequeue()$
      **for** $uv \in A(D)$ with $a_i f(v') \notin E(B(H))$ or
          $vu \in A(D)$ with $f(v) b_{\pi(i)} \notin E(B(H))$ **do**
        Let $(a_i, a_{t_1}), (a_i, a_{t_2}), \ldots, (a_i, a_{t_k}) \in L^2(u, v)$
        Let $P_v \leftarrow \sum_{l=1}^{k} (x_{v,a_{t_l}} - x_{v,a_{t_l+1}})$ and $P_{v,a_{t_j}} \leftarrow \sum_{l=1}^{j} (x_{v,a_{t_l}} - x_{v,a_{t_l+1}}) \,/\, P_v$
        **if** $P_{v,a_{t_j}} < Y \leq P_{v,a_{t_{j+1}}}$ **then** $f(v) \leftarrow a_{t_j}$ and $f(v') \leftarrow b_{\pi(t_j)}$
          $Q.enqueue(v, a_{t_j}, Y_{\lambda+1})$ ▷ $Y_2 = Y_0$
  **for** $u \in B(D)$ **do** if $f(u) = a_t$ then $\chi_{u,a_\iota} = 1$, $1 \leq \iota \leq t$, and $\chi_{u,a_\iota} = 0$, $t < \iota \leq p+1$
  **for** $u' \in B(D)$ **do** if $f(u') = b_t$ then $\chi_{u',b_\iota} = 1$, $1 \leq \iota \leq t$, and $\chi_{u',b_\iota} = 0$, $t < \iota \leq p+1$
  **return** $f$

### 3.5 Correctness and analysis

We first show the correspondence between homomorphisms of $D$ to $H$ and integer solutions of $\mathcal{S}$.

**Lemma 3.3.** *There is a one-to-one correspondence between homomorphisms of $D$ to $H$ and integer solutions of $\mathcal{S}$.*

*Proof.* First, suppose there is a homomorphism $f : V(D) \to V(H)$. For every $v \in V(D)$ when $f(v) = a_t$ then set $x_{v,a_i} = 1$ for all $i \leq t$, and set $x_{v',b_j} = 1$ for all $j \leq \pi(t)$, furthermore, set $x_{v,a_i} = 0$ for all $i > t$ and set $x_{v',b_j} = 0$, for all $j\pi(t)$. Set $x_{v',b_{p+1}} = x_{v,a_{p+1}} = 0$ for all $v, v' \in B(D)$.

Notice that all the variables are non-negative, and we have $x_{v,a_{i+1}} \leq x_{v,a_i}$, and $x_{v',b_{j+1}} \leq x_{v',b_j}$. Observe that by this assignment constrains $(C1), (C2), (C3), (C4), (C5)$, and $(C6)$ are satisfied.

Now for all $u$ and $v$ in $D$ with $f(u) = a_i$ and $f(v) = a_j$ we have that $x_{u,a_i} - x_{u,a_{i+1}} = x_{v,a_j} - x_{v,a_{j+1}} = 1$. Moreover, since $f$ is a homomorphism, we have $(a_i, a_j) \in L^2(u, v)$, and hence, constraint (C7) is also satisfied.

Conversely, from an integer solution for $\mathcal{S}$, we define a homomorphism $f$ from $D$ to $H$ as follows. For every $u \in D$, set $f(u) = a_i$ when $i$ is the largest subscript with $x_{u,a_i} = 1$. Let $uv$ be an arc of $D$ and assume $f(u) = a_i$, $f(v) = a_j$. Note that $x_{u,a_i} - x_{u,a_{i+1}} = x_{v,a_j} - x_{v,a_{j+1}} = 1$ and for all other $s$ we have $x_{v,a_s} - x_{v,a_{s+1}} = 0$. Since constraint (C7) is satisfied,

$$1 \leq \sum_{(a_i,a_s) \in L^2(u,v)} (x_{v,a_s} - x_{v,a_{s+1}}),$$

where $j$ is the only index with $x_{v,a_j} - x_{v,a_{j+1}} \neq 0$ in $\sum_{(a_i,a_s)\in L^2(u,v)} (x_{v,a_s} - x_{v,a_{s+1}})$. Therefore, $(a_i, a_j) \in L^2(u, v)$ and $a_i a_j \in A(H)$. □

It is easy to see that after the execution of Line 7 of Algorithm 1, $f$ is a homomorphism from $B(D)$ to $B(H)$. It is easy to observe the following lemma.

**Lemma 3.4.** *If there is no unstable vertex then the mapping $f$ returned after line 7 of Algorithm 1 is a homomorphism from $D$ to $H$.*

**Lemma 3.5.** *Procedure* SHIFT *runs in polynomial-time and returns a homomorphism from $D$ to $H$.*

### 3.5.1 Proof of Lemma 3.5

This subsection is devoted to the correctness of procedure SHIFT. Some definitions are in order.

**Definition 3.6** ($\overline{L}(X)$)**.** Consider an instance of the list homomorphism problem for digraph $D$, lists $L$ and fixed digraph $H$. For the path $X$ in $D$, let $\overline{L}(X)$ be the sub-digraph of $H$ with the vertex set $\bigcup_{x\in X} L(x)$ and the arc set $\{ab \in A(H) \mid \exists xy \in A(X) \text{ s.t. } a \in L(x), b \in L(y)\}$.

For path $X$ in $D$ we often refer to walk $P \in \overline{L}(X)$; we mean $P$ is congruent to $X$, and the $l$-th element of $P$ is in the list of the $l$-th element of $X$.

**Definition 3.7** ($D \rtimes H^+$)**.** Let $D \rtimes H^+$ be the digraph with vertex set $\{(x, a, b) \mid x \in \mathcal{G}, \text{ and } a, b \in L(x)\}$ and arcs $(x, a, b)(y, c, d)$ so that $(a, c) \in L^2(x, y)$

**Definition 3.8** ($\mathcal{G} \rtimes H^{+2}$)**.** Let $\mathcal{G} \rtimes H^+$ be the digraph with vertex set $\{(x, a, b) \mid x \in \mathcal{G}, \text{ and } a, b \in L(x)\}$ and arcs $(x, a, b)(y, c, d)$ such that :

- There is a minimal path $Y$ from $x$ to $y$ in $D$. Furthermore, $(a, c) \in L^2(x, y)$ (the lists of $D$ with respect to $H$).
- There is a minimal path $P$ in $H$ from $a$ to $c$ and minimal path $Q$ from $b$ to $d$ in $\overline{L}(Y)$ (both congruent with $Y$). There is no path $R$ from $a$ to $d$ in $\overline{L}(Y)$, i.e. $(a, b)(c, d) \in H^{+2}$.

**Remark 3.9.** Let's summarize the construction thus far. We begin with the input hypergraph $\mathcal{G}$ and the target $r$-uniform hypergraph $\mathcal{H}$. From these, we derive the digraphs $D$ and $H$. The system of linear equations $\mathcal{S}$ is then formulated based on $D$ and $H$ for further analysis, particularly concerning the approximation bound. We also have digraph $H^{+2}$, which we use its properties together with semilattice and majority polymorphisms defined on $\mathcal{H}$. We also have construction $D \rtimes H^+$, which is to show the movement of the SHIFT algorithm and allow us to talk about $D$ and $H$ instead of $B(D)$ and $B(H)$. Finally digraph $\mathcal{G} \rtimes H^{+2}$ is used to show that the SHIFT procedure stops after polynomial many times.

We begin by illustrating the properties of digraph $H^{+2}$ together with conservative semilattice polymorphism $\phi$ and conservative majority polymorphism $\psi$ defined in Theorem 2.8. To avoid repeating the word conservative, in the rest of the proof, when we mention polymorphism, we mean a conservative one unless we specify otherwise.

Recall that $(a, b) \in H^{+2}$ is called an invertible pair if $(a, b)$ and $(b, a)$ both belong to the same strong component of $H^{+2}$.

**Observation 3.10.** The following observations for $H^{+2}$ follow from [19]. Here all directed paths such as $(a,b) \rightsquigarrow (c,d)$ are in $H^{+2}$.

1. Suppose $(a,b) \rightsquigarrow (c,d)$. Then by skew-symmetry property, $(d,c) \rightsquigarrow (b,a)$. Moreover, if $\phi(a,b) = a$ then $\phi(c,d) = c$ and if $\psi(a,a,b) = a$ then $\psi(c,c,d) = c$.

2. $(a,b)$ is invertible if and only if $\psi|_{a,b}$ is majority. If $(a,b)$ is invertible $\phi(a,b) = a$ and $\phi(b,a) = b$.

3. Suppose $(a,b)$ is not invertible pair. Then $\phi|_{a,b}$ is semilattice, and $\phi(a,b) = \phi(b,a) = a$ when $(b,a) \rightsquigarrow (a,b)$.

4. Suppose $(a,b) \rightsquigarrow (c,d)$, and $(c,d)$ is invertible. It follows that if $\phi|_{a,b}$ is semilattice then $\phi(a,b) = b$. To see that observe that by skew-symmetry we have $(d,c) \rightsquigarrow (b,a)$, and hence, we have $(a,b) \rightsquigarrow (c,d) \rightsquigarrow (d,c) \rightsquigarrow (b,a)$.

5. If $(a,b)$ and $(c,d)$ are both invertible pairs, and $(a,b) \rightsquigarrow (c,d)$ then both $(a,b)$ and $(c,d)$ belong to the same strong component of $H^{+2}$.

6. If $\psi|_{a,b}$ is the majority, then $\psi$ is the majority on every pair in the same strong component of $H^{+2}$ containing $(a,b)$. Similarly, if $\phi|_{c,d}$ is semilattice then $\phi$ is semilattice on every pair in the same strong component of $H^{+2}$ containing $(c,d)$.

We may assume that SHIFT moves inside $D \rtimes H^+$ starting at $(z, a_r, a_j)$ (or starting at $(z, a_j, a_r)$). This means we change the image of $z'$ from $b_{\pi(j)}$ to $b_{\pi(r)}$. This change potentially affects some in-neighbor $u$ of $z$. Thus, the image of $u$; $f(u)$ where $f(u)a_j \in A(H)$ is changed to some in-neighbor $a_t \in L(u)$ of $a_r$. Therefore, we move from $(z, a_r, a_j)$ to $(u, a_t, f(u))$. Likewise, if we change the image of $z$ to $a_j$, then this change potentially affects some out-neighbor $v$ of $z$. Thus, the image of $v$; $f(v)$, with $a_r f(v) \in A(H)$ is changed to some out-neighbor $a_s \in L(v)$ of $a_j$. Therefore, we move from $(z, a_j, a_r)$ to $(v, a_s, a_{f(v)})$. The above observation allows us to envision the changes of the images of $f$ only for vertices in $D$, and hence, getting rid of Bipartization. We use triple $(y, c, d)$ to keep track of the previous image of the current vertex $y$. This means that when the SBFS visits $(y, c, d)$ in $D \rtimes H^+$ it changes the image of $y$ from $d = f(y)$ to a new value $c$.

**Overview of the proof.** SHIFT goes through strong components of $D \rtimes H^+$. However, a closer look at the paths $P_{x,\mathbf{x}}$ in $D$ and $P_{a,\boldsymbol{\alpha}}$ in $H$ essentially means an image change of a vertex in $P_{x,\mathbf{x}}$ means an image change for a vertex of $x$ in $D$ which is a copy of some vertex in $\mathcal{G}$. Therefore, the important changes are in $\mathcal{G} \rtimes H^{+2}$, which means moving from a vertex $(x,a,b)$ to some $(y,c,d)$ in $\mathcal{G} \rtimes H^{+2}$. These movements are essentially governed by moving inside strong components of $H^{+2}$. When SHIFT reaches a sink strong component (a component that does not have any arc to another component), then it stays on that component until no further changes to the images are necessary.

Let $G$ be a digraph obtained from the projection of the vertices of $\mathcal{G} \rtimes H^{+2}$ on the first coordinate. Note that $G$ has the same vertices as $\mathcal{G}$, and $xy$ is an arc of $G$ whenever $(x,a,b)(y,c,d)$ is an arc of $\mathcal{G} \rtimes H^{+2}$.

To show that procedure SHIFT stops, we need to show that it does not encounter a circuit. A circuit is formed if procedure SHIFT changes the image of some vertex $x \in D \cap \mathcal{G}$ from $a_0$ to $a_1$, and then from $a_1$ to $a_2$, and eventually from some $a_n$ to $a_0$ again. It turns out that all the pairs $(a_1,a_0),(a_2,a_1),\ldots,(a_n,a_{n-1}),(a_0,a_n)$ must be in the same strong component of $H^{+2}$ (for simplicity we use only single subscripts for $a_i$'s). Notice that this circuit occurs in $\overline{L}(X)$ where $X$ is

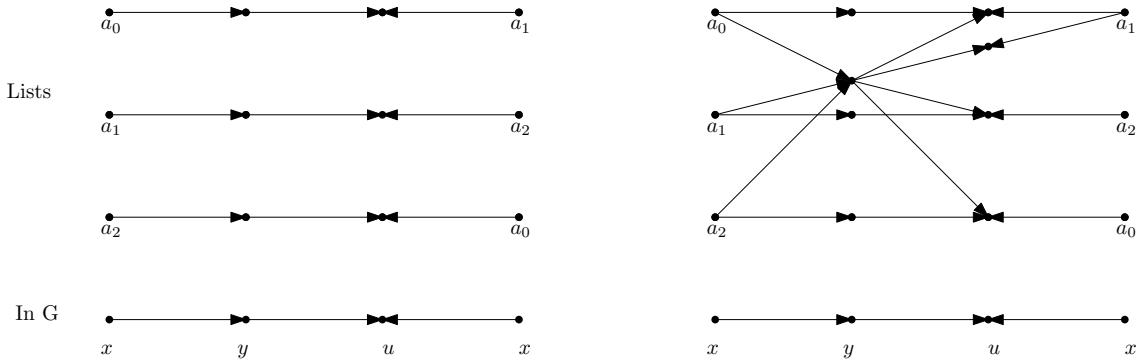


Figure 2: A circuit of length three in list of a cycle of length three in $D$.

a cycle containing vertex $x$ in $G$. Since $a_i \in L(x)$, there is a closed walk $Q_i$, from $a_i$ to $a_i$ in $\overline{L}(X)$. A technical theorem, Theorem 5.4 in [18], gives a structural property of the paths that are obtained from directed paths in $H^{+2}$ from $(a_{i+1}, a_i) \rightsquigarrow (a_{i-1}, a_i)$, which eventually allows us to argue that the SHIFT does not get into a circuit.

We use induction on the total number of elements in lists $L$. Notice that when $\phi(a_r, a_j) = a_r$ then for every $(y, a_t, a_q) \in \mathcal{G} \rtimes H^{+2}$ reachable from $(z, a_r, a_j)$, we have $\phi(a_t, a_q) = a_t$. This follows from Observation 3.10 (1). Therefore, we have the following observation.

**Observation 3.11.** SHIFT visits vertices $(y, c, d)$ where $\phi(c, d) = c$.

We show that procedure SHIFT, starting from $z$ ($z'$) stops after polynomial many steps, and we obtained a partial homomorphism $f$ from $D$ to $H$ under which the weakly connected component of $D$ containing $z$ ($z'$) is homomorphic to $H$. We first show that the procedure SHIFT does not enter a circuit. One might imagine that at the beginning the image of $x$ is shifted from $a_0$ to $a_1$ and then following $X$, and walks $P_0$ and $P_1$ in $\overline{L}(X)$, the image of $x$ is shifted from $a_1$ to $a_2$, and then finally, by following the walks $X$ and $P_n, P_0$ in $\overline{L}(X)$, the image of $x$ is shifted from $a_n$ to $a_0$ (see Figure 2 (left)). It follows that $P_{i+1}$ avoids $P_i$ ($P_{n+1} = P_0$). This means that $P_i$ (from $a_i$ to $a_{i+1}$) avoids $P_{i-1}$ (from $a_{i-1}$ to $a_i$) where both $P_{i-1}$ and $P_i$ are in $\overline{L}(X)$. Therefore, we get pairs

$$C : (a_1, a_0), (a_2, a_1), \ldots, (a_n, a_{n-1}), (a_0, a_n)$$

in $H^{+2}$ so that $(a_i, a_{i-1}) \rightsquigarrow (a_{i+1}, a_i)$ by walks $P_i$ and $P_{i-1}$.

Notice that by definition, all the $(a_{i+1}, a_i)$'s are in the same strong component $S_1$ of $H^{+2}$. We assume that among all the circuits in $\widehat{S}_1$ (a set of pairs that are reachable from $S_1$), $C$ has the shortest length at least three in $\widehat{S}_1$. The length two circuit is treated separately. This allows us to use Theorem 5.4 (1,2,3) in [18], showing that for every $i \neq j$, $P_i$ and $P_j$ avoid each other. Let $Q_{i+1}$ be a walk in $\overline{L}(X)$, from $a_{i+1}$ to $a_{i+1}$ (such a walk exists since $a_{i+1}$ remained in $L(x)$ after the (2,3)-consistency check). Notice that $Q_{i+1}$ is congruent with all of $P_i$'s. We may assume that $Q_{i+1}$ and $P_i$ have a maximum intersection. This means that if $Q_{i+1}$ has a faithful arc at its $l$-th vertex to $P_i$ then $Q_{i+1}$ follows $P_i$ from the $(l+1)$-th vertex. Let $P_{i-1} = b_1, b_2, \ldots, b_r$, $P_i = c_1, c_2, \ldots, c_r$ and $Q_{i+1} = d_1, d_2, \ldots, d_l, c_{l+1}, \ldots, c_r$ where $c_r = a_{i+1}$, $b_r = a_i$, $d_1 = a_{i+1}$, $c_1 = a_i$, and $b_1 = a_{i-1}$.

**Claim 3.12.** *For each $Q_{i+1}$, there is a vertex $d_j \in Q_{i+1}$ so that each $P_\ell$ has a faithful arc to $d_j$. Moreover, there is a vertex $d_t \in Q_{i+1}$, $j \leq t$, such that $d_t$ has faithful arcs to every $P_\ell$.*

*Proof.* We consider two cases depending on whether the pairs in $S_1$ are semilattice pairs or majority pairs.

**Case 1.** First assume that $S_1$ contains semilattice pairs (by Observation 3.10(6), all the pairs in $S_1$ are semilattice). Moreover, by Observation 3.11 for every pair $(a, b) \in S_1$ we have $\phi(a, b) = a$.

If there is no faithful arc from $Q_{i+1}^{-1}$ to $P_i^{-1}$ then $(a_{i+1}, a_i) \rightsquigarrow (a_{i+1}, a_{i-1})$ (using $Q_{i+1}^{-1}$ and $P_{i-1}^{-1}$). Now we have $(a_i, a_{i-1}) \rightsquigarrow (a_{i+1}, a_i) \rightsquigarrow (a_{i+1}, a_{i-1}) \rightsquigarrow (a_{i+2}, a_i)$, which implies a shorter circuit, a contradiction. Thus, let $d_j b_{j-1}$, $j \leq l$, be a faithful arc from $Q_{i+1}^{-1}$ to $P_{i-1}^{-1}$. Now $d_j c_{j-1}$ must be a faithful arc from $Q_{i+1}^{-1}$ to $P_i^{-1}$. Otherwise, $(a_{i+1}, a_i) \rightsquigarrow (a_{i-1}, a_i)$ (using walks $Q_{i+1}[d_1, d_j]P_i^{-1}[b_{j-1}, b_1]$ and $P_i[c_1, c_j]P_i^{-1}[c_{j-1}, c_1]$). On the other hand, we have $(a_i, a_{i-1}) \rightsquigarrow (a_{i+1}, a_i)$. Thus, we have $(a_i, a_{i-1}) \rightsquigarrow (a_{i+1}, a_i) \rightsquigarrow (a_{i-1}, a_i)$, a shorter circuit ( also implying that $(a_i, a_{i-1})$ is not semilattice), and a contradiction.

Similarly, if $d_j c_{j-1}$ is a faithful arc from $Q_{i+1}^{-1}$ to $P_i^{-1}$, then $d_j b_{j-1}$ must be a faithful arc. Now, by applying this argument, for $P_{i-2}$, we conclude that $d_j e_{i-1}$ is a faithful arc where $e_{i-1}$ is the corresponding vertex to $b_{j-1}$ on $P_{i-1}$ (the $(j-1)$-th vertex on $P_{i-1}$). However, by extending this argument to other $P_\ell$, one can show that for $0 \leq \ell \leq n$, $d_j e_\ell$ is a faithful arc where $e_\ell$ is the $(j-1)$-th vertex on $P_\ell$ (see Figure 2(right)).

Now consider $R = Q_{i+1}^{-1}[a_{i+1}, d_j]P_{i-1}^{-1}[b_{j-1}, a_{i-1}]$ and $P_{i+1}^{-1}$. If there is no faithful arc from $P_{i+1}^{-1}[a_{i+2}, e_{i+1}]$ to $Q_{i+1}^{-1}[a_{i+1}, d_j]$, then we have $(a_{i+2}, a_{i+1}) \rightsquigarrow (a_{i+1}, a_{i-1})$ (using $R$ and $P_{i+1}^{-1}$) implying a shorter circuit when the length of the circuit is greater than three. However, when $C$ has length three, we have $(a_0, a_2) \rightsquigarrow (a_2, a_0)$, contradiction $\phi|_{a_0,a_2}$ being semilattice with $\phi(a_0, a_2) = a_0$. Therefore, there is a faithful arc from $P_{i+1}^{-1}[a_{i+2}, e_{i+2}]$ to $Q_{i+1}^{-1}[a_{i+1}, d_j]$, and consequently there is a faithful arc from $Q_{i+1}[d_j, a_{i+1}]$ to $P_{i+1}$. Now assume that $d_t f_{i+1}$ is a faithful arc from $Q_{i+1}[d_j, a_{i+1}]$ to $P_{i+1}$ (note that $t \geq j$). Consider the walks $R = Q_{i+1}^{-1}[a_{i+1}, d_t]P_{i+1}[f_{i+1}, a_{i+2}]$ and $T = P_{i-1}^{-1}[a_i, b_t]P_{i-1}[b_{t+1}, a_i]$. Notice that $d_t b_{t+1}$ ($b_{t+1} \in P_{i-1}$) is a faithful arc; otherwise, we have $(a_{i+1}, a_i) \rightsquigarrow (a_{i+2}, a_i)$ (using $R$ and $T$), a shorter circuit in $\widehat{S_1}$, a contradiction. Now by continuing this argument for $P_{i-2}^{-1}$ and $P_{i-1}^{-1}[a_i, b_{t+1}]d_t P_{i+1}[f_{i+1}, a_{i+2}]$, we conclude that $d_t f_{i-2}$ is a faithful arc where $f_{i-2} \in P_{i-2}$. In general we would have $d_t$ has a faithful to every $P_\ell$, $0 \leq \ell \leq n$.

**Case 2.** All the pairs $(a_{i+1}, a_i)$ are invertible pairs. Thus, by Observation 3.10 (6), $\psi$ is majority on all the pairs in $S_1$. We assumed that the length of $C$ is at least three. There must be a faithful arc from $Q_{i+1}^{-1}$ to $P_{i-1}^{-1}$, otherwise, $(a_{i+1}, a_i) \rightsquigarrow (a_{i+1}, a_{i-1}) \rightsquigarrow (a_{i+2}, a_i)$, and hence, we get a shorter circuit $(a_1, a_0), (a_2, a_1), \ldots, (a_i, a_{i-1}), (a_{i+2}, a_i), (a_{i+3}, a_{i+2}), \ldots, (a_1, a_n)$ in $\widehat{S_1}$. However, by applying the same argument as in Case 1, the claim holds, as long as we assume the length of $C$ is greater than two and we don't make the circuit shorter. So it remains to consider the case where the length of $C$ is three. By symmetry, we may assume $\psi(a_0, a_1, a_2) = a_1$. By applying $\psi$ on walks $P_0, P_1, P_2$ (in $\overline{L}(X)$) from $a_0, a_1, a_2$ to $a_1, a_2, a_0$ (respectively) and the fact that $P_0, P_1, P_2$ pairwise avoid each other, we conclude that $\psi(a_1, a_2, a_0) = a_2$.

Now by applying $\psi$ on $Q_0, P_1$, and $P_2$, we get from $a_0, a_1, a_2$ to $a_0, a_2, a_0$ (respectively). Since $\psi(a_0, a_1, a_2) = a_1$ and $\psi(a_0, a_1, a_0) = a_0$, there must be a faithful arc from $P_1$ to $Q_0$, say arc $cd$. Let us assume $cd$ is the first such an arc. Now by starting from $a_0, a_2, a_1$ and applying $\psi$ on walk

$R_0 = Q_0[a_0, d], R_2 = P_2[a_2, e], R_1 = P_1[a_1, c]d$, we get from $a_0, a_2, a_1$ to $d, e, d$ (respectively) where $e \in P_2$. We show that $e'd$ is a faithful where $e'e$ ($ee'$) is an arc of $P_2$. Suppose this is not the case. Then $(a_2, a_1)$ and $(e, d)$ are in $S_1$ and hence $\psi|_{d,e}$ is majority (since $R_1$ and $R_2$ avoid each other).

Following $R_0, R_1$, and $R_2$ and apply $\psi$ on them, we get from $a_0, a_2, a_1$ to $d, e, d$ and since, $\psi(d, e, d) = d$, and $\psi(a_0, a_2, a_1) = a_2$, there must be a faithful arc from $R_2$ to $R_0$. Let $c'd'$ be such an arc and observe that $d'$ is before $d$. Now by applying $\psi$ on walks $Q_0[a_0, d'], P_1[a_1, f'], P_2[a_2, c']d'$, we get from $a_0, a_1, a_2$ to $d', f', d'$ ($f' \in P_1$ is the corresponding to $d' \in Q_0$). Again similar to the previous argument, since $\psi(a_0, a_1, a_2) = a_1$ and $\psi(d', f', d') = d'$ there must be a faithful arc from $P_1[a_1, f']$ to $Q_0$, contradicting that $cd$ is the first faithful arc from $P_1$ to $Q_0$. This shows that $e'd$ is a faithful arc.

By applying $\psi$ on walks $P_2^{-1}[a_0, e']d$ and $P_0^{-1}[a_1, b']b$ (here $b'b \in P_0$ is the corresponding arc to $cd$) starting from $a_0, a_1, a_2$ to $d, b, d$ (respectively) we get from $\psi(a_0, a_1, a_2) = a_1$ to $\psi(d, b, d) = d$, and hence, $b'd$ must be a faithful arc. This establish the first part of the Claim, i.e. having faithful arcs $b'd$, $cd$ and $e'd$ from $P_0$, $P_1$, $P_2$ to $Q_0$ respectively.

To show that second part of the claim, we start by applying $\psi$ on walks $P_0^{-1}[a_0, b'], P_1^{-1}[a_2, c]$ and $Q_0^{-1}[a_0, q]b'$ from $a_1, a_2, a_0$ to $b', c, b'$ respectively. Thus, we get from $\psi(a_1, a_2, a_0) = a_2$ to $\psi(b', c, b') = b'$. Therefore, there must be a faithful arc from $P^{-1}[a_2, c]$ to $Q^{-1}[a_2, c]$. Thus, there is a faithful arc from $Q_0$ to $P_1$ after $d$, say, $r \in Q_0$. Now analogously to what we have shown to obtain faithful arcs $b'd, cd$ and $e'd$, we conclude that there are faithful arcs from $r$ to $P_0, P_1$ and $P_2$. This complete the proof of the claim.

□

From Claim 3.12, we conclude the following :

- There is a path in $\overline{L}(X)$ from each $a_i$ to each $a_k$, that starts from $a_i$ follows $P_i$ then switches to $Q_{i+1}$ at $d_j$, and following $Q_{i+1}$ and then switches to $P_{k-1}$ from $d_t$ and then gets to $a_k$.

Now suppose the previous image of $X$, has been determined by some $P_t$. Since SHIFT chooses the edges of the new image of $X$ according to the pair consistency constraints, SHIFT starts from $a_{t+1}$, and follows $P_{t+1}$ and then switches to $Q_{i+1}$ and since there is a faithful arc from $Q_{i+1}$ to $P_t$, then SHIFT does not change all images of $X$, and hence, $X$ is mapped to a closed cycle in $\overline{L}(X)$.

**Observation 3.13.** The Algorithms leaves the current component $S_1$.

*Proof.* This is because the algorithm starts from $(x, a_t, a_{t+1})$ and then reaches some $(y, d_j, b)$ (here $d_j \in Q_{i+1}$) and $b \in P_t$, and then to $(y', d_t, b')$ where $b' \in P_t$ and $d_t \in Q_{i+1}$. Since $Q_{t+1}$ has a faithful arc to every $P_\ell$ in $d_t$, $(d_t, b')$ is not in the same strong component as $S_1$. □

Finally, consider the case where the length of $C$ is two. Observe that all pairs in $S_1$ are invertible pairs. This is because we have $(a_1, a_0) \rightsquigarrow (a_0, a_1)$, contradiction to $\phi(a_1, a_0) = a_1$. However, when the length of the circuit is two, SHIFT must take the walk $Q_0$ ($Q_1$) and therefore it does not enter a circuit.

### 3.6 Analyzing the approximation ratio

We now claim that, the expected cost of homomorphism $f$ is at most $|V(H)|$ times the minimum cost of a homomorphism from $D$ to $H$. Let $W$ denote the value of the objective function with the fractional values $x_{u,a_i}, x_{u',b_{\pi(i)}}$, and $W'$ denote the cost of homomorphism returned by Algorithm 1,

i.e., value of the objective function with the final values $\chi_{u,a_i}, \chi_{u',b_{\pi(i)}}$, after rounding and shifting. Also, let $W^*$ be twice the minimum cost of a homomorphism of $D$ to $H$ (see Lemma 3.2). Obviously, $W \leq W^* \leq W'$.

We show that the expected value of $W'$ is at most $|V(H)|$ times $W$. Let us focus on the contribution of one summand, say $\chi_{u,a_t} - \chi_{u,a_{t+1}}$, to the calculation of the cost.

In any integral solution, $\chi_{u,a_t} - \chi_{u,a_{t+1}}$ is either 0 or 1. The probability that $\chi_{u,a_t} - \chi_{u,a_{t+1}}$ contributes to $W'$ is the probability of the event that $\chi_{u,a_t} = 1$ and $\chi_{u,a_{t+1}} = 0$. This can happen in the following situations:

1. $u$ ($u'$) is mapped to $a_t$ ($b_s$) by rounding, and is not shifted away. In other words, we have $\chi_{u,a_t} = 1$ and $\chi_{u,a_{t+1}} = 0$ after rounding, and these values do not change by procedures Shift.
2. $u$ ($u'$) is first mapped to some $a_i$ ($b_j$), by rounding, and then re-mapped to $a_t$ ($b_s$) by procedures Shift.

We next present the following lemma which facilitates the proof of the approximation factor by looking at the contribution of every term of the objective function.

**Lemma 3.14.** *Let $f$ be the homomorphism returned by Algorithm 1. Then for $u, u' \in B(D)$ and $a_t, b_{\pi(t)} \in B(H)$ we have*

$$\mathbb{P}\left[\chi_{u,a_t} = 1, \chi_{u,a_{t+1}} = 0 \text{ i.e. } f(u) = a_t\right] \leq x_{u,a_t} - x_{u,a_{t+1}} \tag{3.1}$$

$$\mathbb{P}\left[\chi_{u',b_{\pi(t)}} = 1, \chi_{u',b_{\pi(t)+1}} = 0 \text{ i.e. } f(u') = b_{\pi(t)}\right] \leq x_{u',b_{\pi(t)}} - x_{u',b_{\pi(t)+1}} \tag{3.2}$$

*Moreover, the expected contribution of each summand, say $c(u,a_t)(\chi_{u,a_t} - \chi_{u,a_{t+1}})$, to the expected cost of $W'$ is at most $|V(H)|c(u,a_t)(x_{u,a_t} - x_{u,a_{t+1}})$.*

*Proof.* After the rounding step using the random variable $X$, we have a homomorphism $f : V(B(G)) \to V(B(H))$, where $f(v) = a_i$ if $x_{u,a_{i+1}} < X \leq x_{u,a_i}$, and $f(u') = b_j$ if $x_{u',b_{j+1}} < X \leq x_{u',b_j}$. Observe that due to constraints (C4) and (C7) each connected component of $B(G)$ is mapped to a connected component of $B(H)$. Since, each connected component of $B(D)$ and $B(H)$ consists of paths of length at most three and ordering is a min-max ordering each edge of $B(D)$ is mapped to an edge of $B(H)$, and hence, $f$ is a homomorphism from $B(D)$ to $B(H)$.

Vertex $u$ is mapped to $a_t$ in two cases. The first case is where $u$ is mapped to $a_t$ by rounding, and is not shifted away. In other words, we have $\chi_{u,a_t} = 1$ and $\chi_{u,a_{t+1}} = 0$ after rounding, and these values do not change by procedure Shift. Hence, for this case we have:

$$\mathbb{P}[f(u) = a_t] \leq \mathbb{P}[x_{u,a_{t+1}} < X \leq x_{u,a_t}] = x_{u,a_t} - x_{u,a_{t+1}}$$

where the first inequality follows from the fact that the probability that the image of $u$ is not changed by Shift procedure is at most 1. Whence, this situation occurs with probability at most $x_{u,a_t} - x_{u,a_{t+1}}$, and the expected contribution of the corresponding summand is at most $c(u,a_t)(x_{u,a_t} - x_{u,a_{t+1}})$.

Second case is where $f(u)$ is set to $a_t$ during procedure Shift. Let $f$ be the mapping from $V(B(D))$ to $V(B(H))$ at the beginning of Shift. Recall that Shift is triggered by an unstable vertex, say $z$, where $f(z) = a_r$, $f(z') = b_{\pi(j)}$ , and $r \neq j$.

We start off with the simplest case where $u$ is unstable and SHIFT stabilizes $u$ by mapping $u$ to $a_t$. This happens when $\chi_{u,a_i} = \chi_{u',b_{\pi(t)}} = 1$, $\chi_{u,a_{i+1}} = \chi_{u',b_{\pi(t)+1}} = 0$ ($f(u) = a_i$, $f(u') = b_{\pi(t)}$) (depending on the value of $\phi$ on $Base(a_i)$ and $Base(a_t)$). Followed by the previous analyses, $u'$ is mapped to $b_{\pi(t)}$ with probability at most $x_{u',b_{\pi(t)}} - x_{u',b_{\pi(t)+1}}$. Moreover, constraint (C6) enforces that $x_{u,a_t} - x_{u,a_{t+1}} = x_{u',b_{\pi(t)}} - x_{u',b_{\pi(t)+1}}$. This therefore yields $u$ is mapped to $a_t$ with probability at most $x_{u,a_t} - x_{u,a_{t+1}}$. There are at most $|V(H)|$ such vertices $a_i$, making $u$ unstable. Thus, we conclude that the expected contribution of $c(u, a_t)$ to $W'$ is at most $|V(H)|c(u, a_t)(x_{u,a_t} - x_{u,a_{t+1}})$ in this case.

In general the image of $u$ is shifted to $a_t$ during SHIFT if there exists an unstable vertex $z$, with $f(z) = a_r, f(z') = b_{\pi(j)}$, $r \neq j$. Suppose SHIFT starts by setting $f(z') = b_{\pi(r)}$ and reaches to $u$ via a walk $Q$ in $D$. The case where $z = u$ was considered in the previous paragraph. Let $Q$ be $z = u^1, u^2, \ldots, u^k, u$ and this is the last time image of $u$ is shifted from a (fixed) $a_i$ and it lands on $a_t$. We use induction on $k$.

For the base case, consider $k = 1$. Let $r$ be the maximum index such that there exists an unstable vertex $z$ with $f(z) = a_r$ and $uz \in A(D)$. Note that this means $f(z') = b_{\pi(j)}$ such that $a_i b_{\pi(j)} \in E(B(H))$ and $r \neq j$. Define events $\mathcal{R}$ and $\mathcal{T}$ as follows:

**Event $\mathcal{R}$: there exists an unstable vertex** $z$ such that $uz$ is an arc of $A(D)$, and $u$ is mapped to $a_i$ and $z$ is mapped to $a_r$ before the call of SHIFT (in the rounding stage according to random variable $X$).

**Event $\mathcal{T}$: the image of $u$ is shifted to $a_t$ from $a_i$.**

We give an upper bound on the probability of event $\mathcal{T}$ given event $\mathcal{R}$. That is, $\mathbb{P}[\mathcal{T} \mid \mathcal{R}]$. Observe that since random variables $X$ and $Y_0$ are independent, these two events are independent, and hence, $\mathbb{P}[\mathcal{T} \mid \mathcal{R}] = \mathbb{P}[\mathcal{T}]\mathbb{P}[\mathcal{R}]$. Define

$$\beta = \max_{w \in \mathcal{U}} x_{w,a_{r+1}}; \quad \mathcal{U} = \{w \mid uw \in A(D) \text{ and } w \text{ is unstable}\}. \tag{3.3}$$

Because of the choice of $a_r$ we have:

$$\mathbb{P}\left[\text{event } \mathcal{R} \text{ happens}\right] \leq \min\left\{x_{u,a_i}, \max_{w \in \mathcal{U}} x_{w,a_r}\right\} - \max\left\{x_{u,a_{i+1}}, \beta\right\} \tag{3.4}$$

$$\leq x_{z,a_r} - \beta \qquad \left(\ z = \operatorname*{argmax}_{w \in \mathcal{U}} x_{w,a_r}\right)$$

$$\leq x_{z,a_r} - x_{z,a_{r+1}} \tag{3.5}$$

Now, consider event $\mathcal{T}$. Let $C = \{a_l \mid (a_l, b_{\pi(r)}) \in L^2(u, z')\}$. Then

$$\mathbb{P}\left[\text{event } \mathcal{T} \text{ happens}\right] = (x_{u,a_t} - x_{u,a_{t+1}}) / \sum_{a_l \in C} (x_{u,a_l} - x_{u,a_{l+1}}) \tag{3.6}$$

Constraint (C6) enforces that $(x_{z,a_r} - x_{z,a_{r+1}}) = (x_{z',b_{\pi(r)}} - x_{z',b_{\pi(r)+1}})$. Followed by constraint (C7-2) we have $(x_{z',b_{\pi(r)}} - x_{z',b_{\pi(r)+1}}) \leq \sum_{a_l \in C} (x_{u,a_l} - x_{u,a_{l+1}})$. Therefore:

$$\mathbb{P}[\mathcal{T} \mid \mathcal{R}] = \mathbb{P}[\mathcal{T}]\mathbb{P}[\mathcal{R}] \tag{3.7}$$

$$\leq (x_{z',b_\pi(r)} - x_{z',b_{\pi(r)+1}})(x_{u,a_t} - x_{u,a_{t+1}}) / (\sum_{a_l \in C} (x_{u,a_l} - x_{u,a_{l+1}})) \tag{3.8}$$

$$\leq (x_{u,a_t} - x_{u,a_{t+1}}) \tag{3.9}$$

Notice that there could exist other vertices $v$ where $uv \in A(D)$, and $v$ was originally unstable, with $f(v) = a_\ell$. If $a_t a_\ell$ is not an arc, then the image of $v$ also changes, and $v$ becomes stable in this round. Now suppose $a_t a_\ell \in A(H)$. Then in the next round when SHIFT stabilizes $v$, the image of $u$ will not change, when the image of $v'$ is changes to $b_{\pi(\ell)}$, and image of $f(v)$ stays as $a_\ell$. Moreover, if the image of $v$ changes to $a_{\pi^{-1}(l)}$ where $f(v') = b_l$, $l \neq \pi(\ell)$, and $a_t a_{\pi^{-1}(l)}$ is not an edge of $H$ then the image of $u$ is shifted away from $a_t$ and the event of mapping $u$ to $a_t$ no longer occurs. These show that only one unstable neighbor of $u$, say $z$, should be considered in computing the probability of shifting the image of $u$ to $a_t$, according to the rules of SHIFT.

Now, for $k > 1$ we should consider the situation where the vertex right before $u$, namely $u^k$, has been shifted, and as a consequence, the image of $u$ should be shifted to $a_t$, which is an in-neighbor (out-neighbor) of $a_l = f(u^k)$. By induction hypothesis on $k$, we have $\mathbb{P}[f(u^k) = a_l] \leq (x_{u^k,a_l} - x_{u^k,a_{l+1}})$.

Let $P_u = \sum_{(a_i,a_l)\in L^2(u,u^k)} (x_{u,a_i} - x_{u,a_{i+1}})$. Suppose we have used random variable $Y_0$ to shift the image of $u^k$ to $f(u^k)$. Now, we shift the image of $u$ from $a_i$ to $a_t$ with probability $P_{u,a_t} = (x_{u,a_t} - x_{u,a_{t+1}})/P_u$, according to random variable $Y_1$, provided that the event that shift the image of $u^k$ to $f(u^k)$ occurs. By induction hypothesis this event is bounded by $(x_{u^k,a_l} - x_{u^k,a_{l+1}})$, and it was solely according to random variable $Y_0$. Since $Y_0$ and $Y_1$ are independent, the probability that $u$ is mapped to $a_t$ is at most :

$$\begin{aligned}\mathbb{P}[f(u) = a_t] &= \mathbb{P}\left[f(u^k) = a_l\right] P_{u,t} \\ &\leq (x_{u^k,a_l} - x_{u^k,a_{l+1}})(x_{u,a_t} - x_{u,a_{t+1}})/P_u \\ &\leq (x_{u,a_t} - x_{u,a_{t+1}}).\end{aligned}$$

The last inequality is due to constraint (C7), stating that $(x_{u^k,a_l} - x_{u^k,a_{l+1}}) \leq P_u$. Now consider a different vertex $v$ where there exists an oriented path $R$ from $v$ to $u$ and $v$ is also unstable with $f(v) = a_r$ and $f(v') = b_{\pi(j)}$ with $j \neq r$ before the process of making $z$ stable. If SBFS goes from $(u, a_t, a_i)$ to $(v, a_s, a_j)$, then it means $v$ becomes stable. On the other hand, if there exists an oriented walk $P$ in $\overline{L}(R^{-1})$ from $a_j$ to $a_t$, during the process of changing the image of $v$ and making it stable, the image of $u$ may changes meaning that the event of mapping $u$ to $a_t$ will not occur. Again, only one unstable neighbor of $u$, say $z$, should be considered in computing the probability of shifting the image of $u$ to $a_t$, according to the rules of SHIFT.

Clearly, during the procedure SHIFT, the image of $u$ can be shifted to $a_t$ because this shift can be initiated from any image $b_l$ for some $y'$ which is not stable. Observe that there are at most $|V(H)| - 1$ of such situations. Therefore, the contribution of $u$ and $a_t$ to the expected value of $W'$ (in the SHIFT stage) is at most $(|V(H)| - 1)c(u, a_t)(x_{u,a_t} - x_{u,a_{t+1}})$ where $(x_{u,a_t} - x_{u,a_{t+1}})$ is the upper bound on the probability provided before.

Overall, considering the rounding and SHIFT, the contribution of $u$ and $a_t$ to the expected value of $W'$ is at most $|V(H)|c(u, a_t)(x_{u,a_t} - x_{u,a_{t+1}})$. □

**Theorem 3.15.** *Algorithm 1 returns a homomorphism with expected cost at most $|V(H)|$ times optimal solution. The algorithm can be derandomized to obtain a deterministic $|V(H)|$-approximation algorithm.*

*Proof.* By Lemma 3.14 and linearity of expectation, for the expected value of $W'$ we heve

$$\begin{aligned}
\mathbb{E}[W'] &= \mathbb{E}\left[\sum_{u,i} c(u,a_i)(\chi_{u,a_i} - \chi_{u,a_{i+1}}) + \sum_{v',j} c(v',b_j)(\chi_{v',b_j} - \chi_{v',b_{j+1}})\right] \\
&= \sum_{u,i} c(u,a_i)\mathbb{E}[\chi_{u,a_i} - \chi_{u,a_{i+1}}] + \sum_{v',j} c(v',b_j)\mathbb{E}[\chi_{v',b_j} - \chi_{v',b_{j+1}}] \\
&\leq |V(H)|(\sum_{u,i} c(u,a_i)(x_{u,a_i} - x_{u,a_{i+1}}) + \sum_{v',j} c(v',b_j)(\chi_{v',b_j} - \chi_{v',b_{j+1}})) \\
&\leq |V(H)|W \leq |V(H)|W^*.
\end{aligned}$$

□

Thus Algorithm 1 outputs a homomorphism whose expected cost is at most $|V(H)|$ times the minimum cost. It can be transformed to a deterministic algorithm as follows. There are only polynomially many values $x_{u,a_t}$ (at most $|V(D)| \cdot |V(H)|$). When $X$ lies anywhere between two such consecutive values, all computations will remain the same. Similarly, when $Y_0$ or $Y_1$ lies anywhere between two consecutive $x_{u,a_t}$s, all the computations remain the same. Moreover, for any given $X$ and $Y_0, Y_1$, the rounding and shifting algorithms can be performed in polynomial time. Thus, we can derandomize the algorithm by trying all the possible values for $X$ and $Y_0, Y_1$ and simply choose the pair that gives us the minimum homomorphism cost. Since the expected value is at most $|V(H)|$ times the minimum cost, this bound also applies to this best solution. The running time of such proceudre is $(|G||H|)^3$.

## References


[1] Sanjeev Arora, László Babai, Jacques Stern, and Z Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *Journal of Computer and System Sciences*, 54(2):317–331, 1997.

[2] Per Austrin. Towards sharp inapproximability for any 2-csp. *SIAM Journal on Computing*, 39(6):2430–2463, 2010.

[3] Amotz Bar-Noy, Mihir Bellare, Magnús M Halldórsson, Hadas Shachnai, and Tami Tamir. On chromatic sums and distributed resource allocation. *Information and Computation*, 140(2):183–202, 1998.

[4] Amey Bhangale and Subhash Khot. Optimal inapproximability of satisfiable k-lin over non-abelian groups. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1615–1628. ACM, 2021.

[5] Amey Bhangale, Subhash Khot, and Dor Minzer. On approximability of satisfiable $k$-csps: I. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 976–988. ACM, 2022.

[6] Amey Bhangale, Subhash Khot, and Dor Minzer. On approximability of satisfiable $k$-csps: II. In *STOC '23: 55th Annual ACM SIGACT Symposium on Theory of Computing, Orlando, Florida, USA, June 20 - 23, 2023*. ACM, 2023.

[7] Amey Bhangale, Subhash Khot, and Dor Minzer. On approximability of satisfiable $k$-csps: III. In *STOC '23: 55th Annual ACM SIGACT Symposium on Theory of Computing, Orlando, Florida, USA, June 20 - 23, 2023*. ACM, 2023.

[8] Andrei A Bulatov. Tractable conservative constraint satisfaction problems. In *Logic in Computer Science (LICS) , 2003. Proceedings. 18th Annual IEEE Symposium on*, pages 321–330. IEEE, 2003.

[9] Nadia Creignou, Sanjeev Khanna, and Madhu Sudan. *Complexity classifications of Boolean constraint satisfaction problems*, volume 7 of *SIAM monographs on discrete mathematics and applications*. SIAM, 2001.

[10] Krzysztof Giaro, Robert Janczewski, Marek Kubale, and Michał Małafiejski. A 27/26-approximation algorithm for the chromatic sum coloring of bipartite graphs. In *International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pages 135–145. Springer, 2002.

[11] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[12] Gregory Gutin, Pavol Hell, Arash Rafiey, and Anders Yeo. A dichotomy for minimum cost graph homomorphisms. *European Journal of Combinatorics*, 29(4):900–911, 2008.

[13] Gregory Gutin, Arash Rafiey, Anders Yeo, and Michael Tso. Level of repair analysis and minimum cost homomorphisms of graphs. *Discrete Applied Mathematics*, 154(6):881–889, 2006.

[14] Magnús M Halldórsson, Guy Kortsarz, and Hadas Shachnai. Minimizing average completion of dedicated tasks and interval graphs. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, pages 114–126. Springer, 2001.

[15] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.

[16] Pavol Hell, Monaldo Mastrolilli, Mayssam Mohammadi Nevisi, and Arash Rafiey. Approximation of minimum cost homomorphisms. In *European Symposium on Algorithms (ESA)*, pages 587–598. Springer, 2012.

[17] Pavol Hell and Jaroslav Nesetril. *Graphs and homomorphisms*. Oxford University Press, 2004.

[18] Pavol Hell, Akbar Rafiey, and Arash Rafiey. Bi-arc digraphs and conservative polymorphisms. *arXiv preprint :1608.03368v5 (2020)*, 2020.

[19] Pavol Hell and Arash Rafiey. The dichotomy of list homomorphisms for digraphs. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms (SODA)*, pages 1703–1713. Society for Industrial and Applied Mathematics, 2011.

[20] Pavol Hell and Arash Rafiey. The dichotomy of minimum cost homomorphism problems for digraphs. *SIAM Journal on Discrete Mathematics*, 26(4):1597–1608, 2012.

[21] Tao Jiang and Douglas B West. Coloring of trees with minimum sum of colors. *Journal of Graph Theory*, 32(4):354–358, 1999.

[22] Peter Jonsson and Gustav Nordh. Introduction to the maximum solution problem. In *Complexity of Constraints*, pages 255–282. Springer, 2008.

[23] Sanjeev Khanna, Madhu Sudan, Luca Trevisan, and David P Williamson. The approximability of constraint satisfaction problems. *SIAM Journal on Computing*, 30(6):1863–1920, 2001.

[24] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM Journal on Computing*, 37(1):319–357, 2007.

[25] Leo G Kroon, Arunabha Sen, Haiyong Deng, and Asim Roy. The optimal cost chromatic partition problem for trees and interval graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 279–292. Springer, 1996.

[26] Ewa Kubicka and Allen J Schwenk. An introduction to chromatic sums. In *Proceedings of the 17th conference on ACM Annual Computer Science Conference*, pages 39–45. ACM, 1989.

[27] Amit Kumar, Rajsekar Manokaran, Madhur Tulsiani, and Nisheeth K. Vishnoi. On lp-based approximability for strict csps. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1560–1573. SIAM, 2011.

[28] Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the max 2-sat and max di-cut problems. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 67–82. Springer, 2002.

[29] Akbar Rafiey, Arash Rafiey, and Thiago Santos. Toward a dichotomy for approximation of h-coloring. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 91:1–91:16, 2019.