On Min-Max Graph Balancing with Strict Negative Correlation Constraints

Ting-Yu Kuo
Dept. of Computer Science, National Yang-Ming Chiao-Tung University, Hsinchu, Taiwan

Yu-Han Chen
Dept. of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan

Andrea Frosini
Dept. of Mathematics and Informatics, University of Florence, Italy

Sun-Yuan Hsieh
Dept. of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan
Dept. of Computer Science and Information Engineering, National Chi-Nan University, Puli, Taiwan

Shi-Chun Tsai
Dept. of Computer Science, National Yang-Ming Chiao-Tung University, Hsinchu, Taiwan

Mong-Jen Kao
Dept. of Computer Science, National Yang-Ming Chiao-Tung University, Hsinchu, Taiwan

Abstract

We consider the min-max graph balancing problem with strict negative correlation (SNC) constraints. The graph balancing problem arises as an equivalent formulation of the classic unrelated machine scheduling problem, where we are given a hypergraph $G = (V, E)$ with vertex-dependent edge weight function $p\colon E \times V \mapsto \mathbb{Z}^{\geq 0}$ that represents the processing time of the edges (jobs). The SNC constraints, which are given as edge subsets $C_1, C_2, \ldots, C_k$, require that the edges in the same subset cannot be assigned to the same vertex at the same time. Under these constraints, the goal is to compute an edge orientation (assignment) that minimizes the maximum workload of the vertices.

In this paper, we conduct a general study on the approximability of this problem. First, we show that, in the presence of SNC constraints, the case with $\max_{e \in E} |e| = \max_i |C_i| = 2$ is the only case for which approximation solutions can be obtained. Further generalization on either direction, e.g., $\max_{e \in E} |e|$ or $\max_i |C_i|$, will directly make computing a feasible solution an NP-complete problem to solve. Then, we present a 2-approximation algorithm for the case with $\max_{e \in E} |e| = \max_i |C_i| = 2$, based on a set of structural simplifications and a tailored assignment LP for this problem. We note that our approach is general and can be applied to similar settings, e.g., scheduling with SNC constraints to minimize the weighted completion time, to obtain similar approximation guarantees.

Further cases are discussed to describe the landscape of the approximability of this prbolem. For the case with $|V| \leq 2$, which is already known to be NP-hard, we present a fully-polynomial time approximation scheme (FPTAS). On the other hand, we show that the problem is at least as hard as vertex cover to approximate when $|V| \geq 3$.

2012 ACM Subject Classification Theory of computation → Scheduling algorithms

Keywords and phrases Unrelated Scheduling, Graph Balancing, Strict Correlation Constraints

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2023.50

Funding *Shi-Chun Tsai*: Supported in part by National Science and Technology Council (NSTC), Taiwan, under Grants 112-2634-F-A49-001-MBK.
Mong-Jen Kao: Supported in part by National Science and Technology Council (NSTC), Taiwan, under Grants 111-2221-E-A49-118-MY3, 112-2628-E-A49-017-MY3, and 112-2634-F-A49-001-MBK.

34th International Symposium on Algorithms and Computation (ISAAC 2023).
Editors: Satoru Iwata and Naonori Kakimura; Article No. 50; pp. 50:1–50:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

In the min-max graph balancing problem with strict negative correlation (SNC) constraints, we are given an edge-weighted hypergraph $G = (V, E)$ with edge weight function $p : E \times V \mapsto \mathbb{Z}^{\geq 0}$ and a collection of edge subsets $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$. An edge orientation (assignment) is a function σ that maps each edge to one of its endpoints, i.e., $\sigma(e) \in e$ for all $e \in E$, and the orientation is said to be feasible if, for any $1 \leq i \leq k$, there exists $e, e' \in C_i$ such that $\sigma(e) \neq \sigma(e')$, i.e., not all edges in C_i are assigned to the same vertex. The workload of a vertex $v \in V$ is defined to be the total weight of the edges assigned to it, i.e., $\sum_{e \in E \text{ s.t. } \sigma(e)=v} p_{e,v}$. The goal of this problem is to compute a feasible edge orientation that minimizes the maximum workload of the vertices.

The graph balancing problem is an equivalent formulation of the classic unrelated machine scheduling problem [22], where the edges in E are interpreted as jobs, the vertices V are the machines, and the weights of edges are the processing times of the jobs. In the following, we start with an introduction on the unrelated scheduling problem.

Lenstra et al. [22] presented an elegant LP-rounding scheme that exploits the extreme point structure and obtained a 2-approximation for the unrelated scheduling problem. They also showed that $(1.5 - \epsilon)$-approximation for any $\epsilon > 0$ is NP-hard to obtain. Since then, there has been no significant progress on the upper-bound nor lower-bound for this problem, and closing the gap is known as a major open problems in this field for over 30 years [26, 29].

It is worth noting that, even the strongest LP formulation ever known for this problem, i.e., the configuration LP [27, 4], has an integrality gap of 2 for this problem [28]. Due to the above reasons, subsequent research has mostly focused on restricted cases of the problem.

An important subcase that is widely considered in the literature is the restricted assignment case, which considers the vertex-independent edge weight function $p\colon E \mapsto \mathbb{Z}^{\geq 0}$. Svensson [27] showed that the configuration-LP has an integrality gap at most $33/17 \approx 1.9412$ for this problem. This bound was later improved to $11/6 \approx 1.833$ by Jansen and Rohwedder [21]. In terms of approximation guarantees, Chakrabarty et al. [9] showed that, when there are only two different types of edge weights, a $(2 - \delta)$-approximation can be obtained for some small fixed constant $\delta > 0$.

For restricted assignment case without hyperedges, i.e. $p\colon E \mapsto \mathbb{Z}^{\geq 0}$ and $\max_{e \in E} |e| = 2$, Ebenlendr et al. [16] presented a 1.75-approximation algorithm. They also showed that, even for this case, a $(1.5 - \epsilon)$-approximation is still NP-hard to obtain. Moreover, the hardness result in [16] holds when there are only two different types of edge weights. For this seemingly simple case, a 1.5-approximation can be obtained [19, 23, 10]. Interestingly, this is the only nontrivial special case of unrelated scheduling for which the exact approximability is known.

Our motivation for studying the SNC constraints originates from the growing attention on the pairwise negative correlation between jobs to surpass the long-standing guarantees for job scheduling to minimize the weighted completion time [3, 5]. In our setting, we consider the extreme case for which the negative correlation between jobs in the same group is one.

In general, the presence of SNC constraints makes the problem much harder to consider. Consider the *constraint graph* $G_{\mathcal{C}} := (E, \mathcal{C})$ with the edges in E being the vertices and the constraints in $\mathcal{C}$ being the hyperedges. Even for the case that G is a complete hypergraph, i.e., $e = V$ for all $e \in E$, determining whether or not G has a feasible edge orientation is already equivalent to the problem of determining whether or not $G_{\mathcal{C}}$ has a $|V|$-coloring such that no constraint in $\mathcal{C}$ is monochromatic. As graph coloring is NP-hard, determining the existence of feasible edge orientation in the presence of SNC constraints is in general NP-hard.

There are essentially two directions to bypass the inherent hardness of the SNC constraints. The first one is to assume that a feasible coloring for $G_{\mathcal{C}}$ is given in advance, e.g., [7], and the other is to consider restricted classes of $G_{\mathcal{C}}$ for which a feasible orientation is polynomial-time computable, e.g., [20, 13, 24, 25]. Notably, most of these works assumed restricted assignment case with complete hypergraph, i.e., $p\colon E \mapsto \mathbb{Z}^{\geq 0}$ and $e = V$ for all $e \in E$, which is known as the identical machine scheduling case in the literature, with special constraint graphs with $|C_i| = 2$ for all $1 \leq i \leq k$.

In the following we introduce the above results in more detail. Bodlaender et al. [7] showed that, when a χ-coloring for the constraint graph $G_{\mathcal{C}}$ is given in advance, a $(\chi + 2)/2$-approximation can be obtained when $\chi \leq |V| - 1$, and a 3-approximation can be obtained when $\chi \leq |V|/2 + 1$. This is achieved by partitioning the vertices into χ groups in a way such that no SNC constraints exist for each group. Different approximation guarantees are obtained, based on different heuristics to distribute the number of vertices for each color.

Jansen et al. [20] considered the case for which $G_{\mathcal{C}}$ is a complete multipartite graph, i.e., the edges in E are partitioned into multiple groups, and each vertex must handle edges that are within the same group. For this case, they provided a polynomial-time approximation scheme (PTAS). Pikies et al. [25] further considered the unrelated scheduling case and gave a $(1 + \epsilon)\overline{p}$-approximation for any $\epsilon > 0$, where $\overline{p} = \max_{e,v} p_{e,v}/\min_{e,v} p_{e,v}$ is the maximum ratio between the edge weights. This is done by ignoring the edge weights and applying the algorithm of Jansen et al. [20]. Surprisingly, this straightforward algorithm is proven to be tight. They also showed that, even when $G_{\mathcal{C}}$ is complete bipartite, an $O(n^b \overline{p}^{1-c})$-approximation is NP-hard to obtain for any $b, c > 0$.

Das and Wiese [13] considered the case for which $G_{\mathcal{C}}$ is a collection of cliques, i.e., none of the edges from the same clique can be assigned to the same machine. For this case, they achieved a PTAS for identical machine scheduling. For unrelated machine scheduling, they proved a $(\log n)^{1/4}$-inapproximability unless $\mathsf{NP} \subseteq \mathsf{ZPTIME}(2^{(\log n)^{O(1)}})$. For the positive side, Page and Solis-Oba [24] provided a b-approximation, where b is the number of cliques in $G_{\mathcal{C}}$. They also gave a $b/2$-approximation for the restricted case that $\max_{e \in E} |e| = 2$.

Further related works

A problem directly related to min-max graph balancing is the max-min fair allocation, for which the goal is to maximize the minimum workload of the machines under the same set of inputs [6, 4]. For the unrelated scheduling case, i.e., $p\colon E \times V \mapsto \mathbb{Z}^{\geq 0}$, it is known that, $(2 - \epsilon)$-approximation for any $\epsilon > 0$ is NP-hard to obtain [6]. For any $\epsilon = \Omega(\log\log n/\log n)$, Chakrabarty et al. [8] provided an $O(n^\epsilon)$-approximation in $O(n^{1/\epsilon})$-time. Furthermore, it is known that, the integrality gap of configuration LP is $\Omega(\sqrt{|V|})$ even when $|E| = O(|V|)$ [4].

For the restricted assignment case, i.e., $p\colon E \mapsto \mathbb{Z}^{\geq 0}$, it is known that $(2 - \epsilon)$-approximation is still NP-hard to obtain [6]. Bansal and Sviridenko [4] presented an $O(\log\log m/\log\log\log m)$-approximation based on rounding the configuration LP. In a series of follow-up works [17, 2, 12] and a very recent work due to Haxell and Szabó [18], the integrality gap of configuration LP for this case is narrowed down to 3.534.

The first constant factor approximation for this case was obtained by Annamalai et al. [1], which introduced the concept of lazy updates on the algorithm of [2] for polynomial-time termination. The approximation guarantee was further improved to 6 [11, 14] and then 4 [12], by further deriving more structures for the local search algorithm. For the case for which $\max_e |e| = 2$, a tight 2-approximation can be obtained [8]. Notably, this is also the only special case for which the exact approximability is known for max-min fair allocation.

Our Results and Contributions

In this paper, we study the complexity of unrelated graph balancing problem with SNC constraints and provide a clear landscape on the approximability of this problem with respect to different structures of input graphs. In contrast to the previous works, e.g., [20, 25, 13, 24], which mostly considered SNC constraints with special structures, we always keep SNC constraints in its most general form and discuss the complexity of the problem.

First, we show that, in the most general setting, either $\max_{e\in E}|e| \geq 3$ or $\max_i |C_i| \geq 3$ directly makes it NP-hard to even determine the existence of a feasible solution for the input instance. Hence, the case that $\max_{e\in E}|e| = \max_i |C_i| = 2$ is the only case for which approximation solutions can be obtained in terms of polynomial-time computations.

Even for the case $\max_{e\in E}|e| = \max_i |C_i| = 2$, determining the feasibility of the input instance is still not a trivial task to accomplish. For this, we provide a characterization of infeasible instances that can be checked in polynomial-time. This is done by transforming the problem into an *implication graph* between the assignments.

Then, we present a 2-approximation algorithm for the case with $\max_{e\in E}|e| = \max_i |C_i| = 2$. Our ingredient for this part is LP-rounding that further exploits the implication between assignments. We transform the concept into a directed acyclic graph (DAG), for which we design a specific assignment LP. We provide a threshold-based rounding, which follows the topological ordering of the DAG. The feasibility of the rounded solution is then ensured by the DAG structure.

Note that, even when there is no SNC constraint, the ratio of 2 is still the best approximation guarantee known for the case with $\max_{e\in E}|e| = 2$. We also remark that, our approach is general and can be directly applied to similar problems, e.g., scheduling with SNC constraints to minimize the weighted completion time, to obtain a 2-approximation guarantee.

It is also worth noting that, the techniques by Ebenlendr et al. [16] and Chakrabarty et al. [8], which are used to obtain approximation results for the restricted assignment case with no SNC constraints, do not seem to be applicable here. A key step in their rounding algorithms is to fractionally-round a cycle for G while keeping the remaining assignments unchanged. With the SNC constraints in place, such a rounding step is not guaranteed.

To compose a complete landscape for this problem, further special cases for G are discussed. For the case when $|V| \leq 2$, we show that a fully polynomial-time approximation scheme (FPTAS) can be obtained, based on a pseudo-polynomial time dynamic programming algorithm. Note that this case already contains the partition problem as its special case and is NP-hard to solve. On the other hand, we show that the problem is at least as hard as vertex cover to approximate when $|V| \geq 3$. Hence, assuming the unique game conjecture, our approximation result is already tight for this case.

Organization of this Paper

The rest of this paper is organized as follows. In Section 2, we provide the hardness result when $\max_{e\in E}|e| \geq 3$ or $\max_i |C_i| \geq 3$. In Section 3, we present a characterization for infeasible instances and our 2-approximation algorithm. We provide our FPTAS for $|V| \leq 2$ and the hardness results for $|V| \geq 3$ in Section 4.

2 Preliminaries

In the min-max SNC-graph balancing problem, we are given a tuple $\Psi = (G = (V, E),\ p,\ \mathcal{C})$, where $G = (V, E)$ is a hypergraph, $p : E \times V \mapsto \mathbb{Z}^{\geq 0}$ is a vertex-dependent edge weight function, and $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ is a collection of edge subsets that is referred to as the SNC constraints.

An edge orientation (assignment) is a function σ that maps each edge to one of its endpoints, i.e., $\sigma(e) \in e$ for all $e \in E$, and the orientation σ is said to be feasible if, for any $1 \le i \le k$, there exists $e, e' \in C_i$ such that $\sigma(e) \neq \sigma(e')$, i.e., not all edges in C_i are assigned to the same vertex. The workload of a vertex $v \in V$ w.r.t. σ is defined to be the total weight of the edges assigned to it, i.e., $\sum_{e \in E \text{ s.t. } \sigma(e)=v} p_{e,v}$. The goal of this problem is to compute a feasible edge orientation that minimizes the maximum workload of the vertices.

Let $\Psi = (G = (V, E),\ p,\ \mathcal{C})$ be an instance of the min-max SNC-graph balancing. For any edge orientation σ, we will use $\sigma^{-1}(v) := \{e \in E \mid \sigma(e) = v\}$ for any $v \in V$ to denote the set of edges that are assigned to v.

We say that Ψ is an (α, β)-instance if $\max_{e \in E} |e| = \alpha$ and $\max_{C \in \mathcal{C}} |C| = \beta$. In the min-max (α, β)-SNC graph balancing problem, we assume that Ψ is an (α, β)-instance.

Complexity of Min-Max (α, β)-SNC Graph Balancing

In the following, we show that, when $\max(\alpha, \beta) \ge 3$, determining whether or not an (α, β) instance has a feasible solution is already an NP-hard problem. Hence, min-max $(2, 2)$-SNC graph balancing is the only case for which an approximation solution can be obtained in terms of polynomial-time computations.

For this, we consider the cases $\alpha \ge 3$ and $\beta \ge 3$ separately and construct NP-hard reductions for them. We note that, as the weight function p plays no role in determining the feasibility of the instance, we will omit the construction detail for p.

First, for the case $\alpha \ge 3$, we make a reduction from the 3-SAT problem. Let $\varphi = \{c_1, c_2, \ldots, c_m\}$ be a set of m clauses over n variables $x_1, \ldots, x_n$. We construct an instance $\Psi = (G = (V, E), p, \mathcal{C})$ with $\max_{e \in E} |e| \le 3$ as follows. For each variable x_i, we create two literal vertices v_{x_i} and $v_{\neg x_i}$ and an edge $e_{x_i} = \{v_{x_i}, v_{\neg x_i}\}$. Intuitively, this edge is supposed to be oriented to the negated value of x_i in a satisfying assignment, i.e., e_{x_i} should be oriented to $v_{\neg x_i}$ if x_i is true in a satisfying assignment and vice versa.

For each clause c_j, we construct a hyperedge e_{c_j} which contains the three literal vertices that c_j contains. Furthermore, for each clause c_j and each variable, say, x_i, that appears in c_j, we create an SNC constraint $C_{j,i} = \{e_{c_j}, e_{x_i}\}$. Intuitively, the hyperedge e_{c_j} for each clause c_j is supposed to be oriented to one of the literals that is true in a satisfying assignment, and the consistency between the orientations of the variables and clauses is provided by the SNC constraints we created. We have the following lemma.

▶ **Lemma 1.** *φ is satisfiable if and only if there exists a feasible orientation for Ψ.*

For the case $\beta \ge 3$, we make a reduction from 3-uniform hypergraph 2-coloring [15]. We show that, the problem of computing a feasible orientation for $(2, 3)$-SNC graph balancing already contains the 3-uniform hypergraph 2-coloring problem as one of its special cases.

Recall that, in the 3-uniform hypergraph 2-coloring problem, we are given a 3-uniform hypergraph $G = (V, E)$ and the goal is to decide if there exists a 2-coloring of the vertices in V such that no edge is monochromatic.

We construct an instance $\Psi' = (G' = (V', E'), p, \mathcal{C}')$ with $\max_{C \in \mathcal{C}'} |C| = 3$ as follows. The vertex set V' consists of two vertices $v^{(0)}, v^{(1)}$ which correspond to the colors we are using. For each vertex $v \in V$, we create an edge e_v in E' with end-points $v^{(0)}, v^{(1)}$. Note that this creates multi-edges between $v^{(0)}$ and $v^{(1)}$ in G'. Intuitively, the orientation of e_v corresponds to the color of vertex v in a valid 2-coloring.

For each 3-uniform hyperedge $e \in E$, say, with endpoints $u, v, w \in V$, we create an SNC constraint $C_e := \{e_u, e_v, e_w\}$ in $\mathcal{C}'$. Intuitively, the SNC constraint requires that not all endpoints of e are assigned to the same vertex, and this models the feasibility of the 2-coloring for G. The following lemma establishes the correctness of the reduction.

▶ **Lemma 2.** *G is 2-colorable if and only if Ψ' is feasible.*

By Lemma 1 and Lemma 2, we obtain the following theorem.

▶ **Theorem 3.** *When $\max(\alpha, \beta) \geq 3$, it is NP-hard to determine the feasibility of (α, β)-instances for the min-max SNC graph balancing problem.*

3 Min-Max $(2, 2)$-SNC Graph Balancing

In this section, we consider the min-max $(2, 2)$-SNC graph balancing problem. First, we present a characterization of feasible instances that can be tested in polynomial-time. Then, we introduce a set of structural properties and modifications on the instance followed with an assignment LP and obtain a 2-approximation for feasible instances of this problem.

Let $\Psi = (G = (V, E),\ p,\ \mathcal{C})$ be an instance of min-max $(2, 2)$-SNC graph balancing, i.e., $|e| \leq 2$ for all $e \in E$ and $|C| = 2$ for all $C \in \mathcal{C}$. To simply the notation, for any $e \in E$ and any $v \in e$, we will use $e \setminus v$ to denote the endpoint of e other than v. Furthermore, $e \setminus v$ is defined to be ϕ if v is the only endpoint of e, i.e., e is a self-loop.

3.1 The Implication Graph and a Feasibility Characterization

In the following, we first define the concept of *implication graph* H for Ψ and a set of *bad implications* in the implication graph H. Then we show that Ψ is feasible if and only if there exists no bad implication in H.

Consider any SNC constraint $\{e, e'\} \in \mathcal{C}$. If v is a common endpoint of e and e', i.e., $v \in e \cap e'$, and if e is already assigned to v, then e' must not be assigned to v in any feasible assignment. In other words, e' must be assigned to $e' \setminus v$. In this scenario, we say that the assignment of e to v *implies* the assignment of e' to $e' \setminus v$.

The above observation defines the directed implication graph $H = (V_H, E_H)$. The vertex set V_H consists of two types of nodes, namely,

- $u_{e,v}$ for each $e \in E$ and each $v \in e$, and
- $u_{e,\phi}$ for each $e \in E$ with $|e| = 1$.

Intuitively, we construct H in a way such that, if $u_{e,v}$ is implied by a directed arc in E_H, then e is supposed to be assigned to v in any feasible assignment. Furthermore, if $u_{e,\phi}$ is implied by an arc, then the instance Ψ is infeasible.

The directed arcs in E_H are defined as follows. For each SNC constraint $\{e, e'\} \in \mathcal{C}$ and each $v \in e \cap e'$, we create two arcs: One from $u_{e,v}$ to $u_{e',e'\setminus v}$ and the other from $u_{e',v}$ to $u_{e,e\setminus v}$. Intuitively, the two arcs indicate that, if one of e or e' is assigned to v, then the other edge must be assigned to the vertex other than v.

Following the above concept, we use $u_{e,v} \overset{+}{\rightarrow} u_{e',v'}$ to denote the scenario where there exists a path of nonzero length from $u_{e,v}$ to $u_{e',v'}$ in H. If both $u_{e,v} \overset{+}{\rightarrow} u_{e',v'}$ and $u_{e',v'} \overset{+}{\rightarrow} u_{e,v}$, then we write $u_{e,v} \overset{+}{\leftrightarrow} u_{e',v'}$. Intuitively, if $u_{e,v} \overset{+}{\leftrightarrow} u_{e',v'}$, then there exists a cycle that passes both $u_{e,v}$ and $u_{e',v'}$. Furthermore, the assignment of any edge on the nodes of this cycle will uniquely determine the assignments of all the edges on the nodes of the same cycle.

▶ **Definition 4** (Bad Implication)**.** *The following chains of implications are considered* bad.

1. *There exists a cycle in H that passes through both $u_{e,v}$ and $u_{e,v'}$ for some $e = \{v, v'\} \in E$, i.e., $u_{e,v} \overset{+}{\leftrightarrow} u_{e,v'}$ for some $e = \{v, v'\} \in E$.*
2. *$u_{e,\phi}$ is implied by $u_{e,v}$ for some $e = \{v\} \in E$, i.e., $u_{e,v} \overset{+}{\rightarrow} u_{e,\phi}$ for some $e = \{v\} \in E$.*

Clearly, the instance Ψ is infeasible if $u_{e,v}$ and $u_{e,v'}$ imply each other for some $e = \{v, v'\}$ or $u_{e,\phi}$ is implied by $u_{e,v}$ for some $e = \{v\} \in E$. The following lemma, on the contrary, shows that the obvious necessary condition is also sufficient.

▶ **Lemma 5.** *Ψ has a feasible orientation if and only if there is no bad implication in H.*

Although Lemma 5 can be proved directly, we chose to prove it in an implicit way. We show in the following sections that, when there is no bad implication in H, a 2-approximation for Ψ can be computed based on LP-rounding. This completes the proof of Lemma 5.

We also note that, the existence of bad implications can be tested in polynomial-time by simple graph traversal in H. We obtain the following theorem.

▶ **Theorem 6.** *The feasibility of Ψ can be tested in polynomial-time.*

3.2 Unique Edge Orientation and Strongly Connected Components

In the following, we assume that no bad implication exists in the implication graph H. We further simplify the structure of H by identifying

1. edges whose orientations can be uniquely determined, and
2. edges whose orientations are implied by each other.

In the former case, the edges will be assigned directly as *dedicated workloads* that each vertex in V possesses. The latter case corresponds to strongly connected components (SCCs) in H to be contracted and treated as a single vertex. When this process ends, we obtain a simplified implication graph $H'' = (V_{H''}, E_{H''})$, which is directed acyclic, and a dedicated workload function $q\colon V \mapsto \mathbb{Z}^{\geq 0}$ of the vertices. In the following we describe the details.

Unique Edge Orientation

Observe that, the assignment of an edge $e \in E$ with $v \in e$ can be uniquely determined if one of the following two cases holds.

- $e = \{v\}$, i.e., e is a self-loop. Then e must be assigned to v.
- $u_{e,v'} \xrightarrow{+} u_{e,v}$, where $e = \{v, v'\}$. In this case, it also follows that e must be assigned to v.

In addition, provided that the edge e is to be assigned to v, all the nodes (assignments) that are further implied by $u_{e,v}$ in H must be *realized* as well. On the other hand, the opposite direction of the realized assignments, e.g., $u_{e,e\setminus v}$, must never be made and should be removed from the implication graph H.

In the following, we describe a unifying approach to handle the above two cases. We start with a zero dedicated workload function $q \leftarrow 0$ and repeat the following steps while there exists some $v \in e \in E$ such that either $|e| = 1$ or $u_{e,e\setminus v} \xrightarrow{+} u_{e,v}$.

Inside the main while loop, we pick one such $v \in e \in E$ and do the following. Let

$$A \;\leftarrow\; \{u_{e,v}\} \cup \left\{\, \ell \in V_H \;\mid\; u_{e,v} \xrightarrow{+} \ell \,\right\}$$

be the set of nodes (assignments) in H that are implied by $u_{e,v}$, i.e., the set of nodes reachable from $u_{e,v}$. Intuitively, the assignments in A must be realized as well. On the contrary, let

$$B \;\leftarrow\; \left\{\, u_{e',e'\setminus v'} \;\mid\; u_{e',v'} \in A \,\right\}$$

be the set of nodes that make the opposite directions of assignments to the nodes in A. Intuitively, the assignments in B must not be realized.

Then, we do the following updates. For each node, say, $u_{e',v'} \in A$, we assign e' to v' and add $p_{e',v'}$ to $q_{v'}$ as dedicated loads of v' accordingly. Then we remove both A and B from H and proceed to the next iteration until there exists no $v \in e \in E$ with $|e| = 1$ or $u_{e,e\setminus v} \xrightarrow{+} u_{e,v}$. In the following, we use Algorithm A to denote the above process.

In the following, we show that, for any $e \in E$ and any $v \in e$, the two nodes $u_{e,v}$ and $u_{e,e\setminus v}$ cannot belong to A at the same time. Hence, the concepts of A and B in Algorithm A are well-defined. We begin with the following structural lemma for H. Intuitively, it provides a reversed symmetric property for the *conjugating pair of nodes* in H in that, whenever $u_{e,v} \xrightarrow{+} u_{e',v'}$ for some $u_{e,v}, u_{e',v'} \in V_H$, their conjugating partners, $u_{e,e\setminus v}$ and $u_{e',e'\setminus v'}$, must have a reversed implication relation $u_{e',e'\setminus v'} \xrightarrow{+} u_{e,e\setminus v}$.

▶ **Lemma 7.** *Let $e, e' \in E$ with $v \in e, v' \in e'$. If $u_{e,v} \xrightarrow{+} u_{e',v'}$, then $u_{e',e'\setminus v'} \xrightarrow{+} u_{e,e\setminus v}$.*

Proof. We prove by induction on the length n of the shortest path from $u_{e,v}$ to $u_{e',v'}$. If $n = 1$, then by the definition of H, we have $\{e, e'\} \in \mathcal{C}$ and $v \in e \cap e'$, and $v = e' \setminus v'$. Hence, when e' is assigned to v, e must be assigned to $e \setminus v$. Therefore we have $u_{e',e'\setminus v'} \xrightarrow{+} u_{e,e\setminus v}$.

Assume that the statement holds when the length of the shortest path from $u_{e,v}$ to $u_{e',v'}$ is at most n. Then for the length $n+1$, pick an arbitrary intermediate vertex ℓ on the shortest path from $u_{e,v}$ to $u_{e',v'}$. That is to say, $u_{e,v} \xrightarrow{+} \ell$ and $\ell \xrightarrow{+} u_{e',v'}$. It follows that the lengths of both subpaths is at most n. So by assumption, we have $u_{e',e'\setminus v'} \xrightarrow{+} \ell' \xrightarrow{+} u_{e\setminus v}$, where ℓ' is the conjugating pair of ℓ. This proves the lemma. ◀

The following lemma shows that the concepts of A and B in Algorithm A are well-defined.

▶ **Lemma 8.** *For any $v \in e \in E$, $u_{e,v} \in A$ implies that $u_{e,e\setminus v} \notin A$.*

Proof. Consider any iteration in Algorithm A. Let (v^*, e^*), where $v^* \in e^* \in E$, denote the pair that is selected in the beginning of the iteration such that either $|e^*| = 1$ or $u_{e^*,e^*\setminus v^*} \xrightarrow{+} u_{e^*,v^*}$.

Assume for contradiction that, for some $v \in e \in E$, both $u_{e,v}$ and $u_{e,e\setminus v}$ are in A. Depending on whether or not $e = e^*$, we distinguish two cases and show that they both lead to bad implications in H. Note that this will be a contradiction to our assumption in H.

- $e = e^*$ and $v = v^*$, i.e., (e, v) is the pair chosen in the beginning of this iteration. In this case, since $u_{e^*,e^*\setminus v^*} = u_{e,e\setminus v} \in A$, we have $u_{e^*,v^*} \xrightarrow{+} u_{e^*,e^*\setminus v^*}$, which is a bad implication.
- Assume that $e \neq e^*$. Since both $u_{e,v}, u_{e,e\setminus v} \in A$, it follows that

  $$u_{e^*,v^*} \xrightarrow{+} u_{e,v} \quad \text{and} \quad u_{e^*,v^*} \xrightarrow{+} u_{e,e\setminus v} \tag{1}$$

  hold at the same time. By Lemma 7, this implies that

  $$u_{e,e\setminus v} \xrightarrow{+} u_{e^*,e^*\setminus v^*} \quad \text{and} \quad u_{e,v} \xrightarrow{+} u_{e^*,e^*\setminus v^*} \tag{2}$$

  hold at the same time. We further consider the two subcases for which $|e^*| = 1$ or not.

  - If $|e^*| \neq 1$, then we have $u_{e^*,e^*\setminus v^*} \xrightarrow{+} u_{e^*,v^*}$ by the condition we pick at the beginning of the while loop. Then we have $u_{e,e\setminus v} \xrightarrow{+} u_{e^*,e^*\setminus v^*} \xrightarrow{+} u_{e^*,v^*} \xrightarrow{+} u_{e,v}$ by (1) and (2), which is bad.
  - If $|e^*| = 1$, then $u_{e^*,v^*} \xrightarrow{+} u_{e,v} \xrightarrow{+} u_{e^*,e^*\setminus v^*}$ is a bad implication since $u_{e^*,e^*\setminus v^*} = u_{e^*,\phi}$.

In all cases, it leads to a bad implication, which is a contradiction to the assumption that Ψ is a feasible instance. This proves the lemma. ◀

By Algorithm A, we assume in the following that there are no self-loops in G and for any $e = \{v, v'\} \in E$, none of $u_{e,v}$ or $u_{e,v'}$ imply each other. Furthermore, we have a dedicated workload function q for the vertices in V.

Handling the Strongly Connected Components

Consider the case that $\ell \overset{+}{\Leftrightarrow} \ell'$ for some $\ell, \ell' \in V_H$. Clearly this corresponds to a directed cycle of implications, say, C, in H and constitutes as part of a strongly connected component (SCC), say, C'. It follows that, for any node on the cycle, say $u_{e,v} \in C$, if the orientation of e is determined, then the orientation of all the remaining edges to which the nodes on the cycle correspond is also determined.

In fact, it is straightforward to verify that, the orientation of all the edges in the component C' are mutually bound to each other. From this observation, the whole component C' can be treated as a single node in the implication graph H, since the assignments of all the edges on the nodes of this component are bound together.

In the following we formally define this concept. Let H' be the updated implication graph after Algorithm A is applied and $C'_1, C'_2, \ldots, C'_k$ be the SCCs we have in H'.

Define the contracted implication graph $H'' = (V_{H''}, E_{H''})$ as follows. For each $1 \le i \le k$, we have a vertex v_i in $V_{H''}$ that represents the component C'_i. For any $1 \le i, j \le k$, we draw an arc (v_i, v_j) in $E_{H''}$ if there is an arc (ℓ, ℓ') that connects some $\ell \in C'_i$ to some $\ell' \in C'_j$.

Intuitively, the graph H'' is obtained by contracting each SCC in H' into a single vertex. Since there is a one-to-one correspondence between SCCs in H' and the vertices in $V_{H''}$, we will use $\delta(s)$ for any $s \in V_{H''}$ to denote the SCC to which s corresponds in H'. The following lemma is straightforward to verify.

▶ **Lemma 9.** *H'' is acyclic.*

The following structural lemma for SCCs in H', obtained from Lemma 7, shows that SCCs in H' also form conjugating pairs, regardless of their sizes.

▶ **Lemma 10.** *For any $e, e' \in E$ with $v \in e$, $v' \in e'$, if $u_{e,v}$ and $u_{e',v'}$ belong to the same SCC, then $u_{e,e\setminus v}$ and $u_{e',e'\setminus v'}$ must belong to the same SCC as well.*

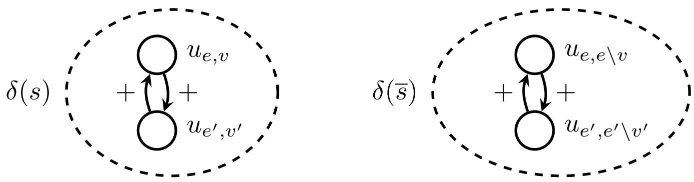

Figure 1 An illustration of the definition of conjugating pairs in $V_{H''}$.

Lemma 10 allows the concept of conjugation for SCCs to be defined. Formally, for any $s \in V_{H''}$ and any $u_{e,v} \in \delta(s)$, define $\overline{s}$ to be the vertex in $V_{H''}$ such that $\delta(\overline{s})$ contains the node $u_{e,e\setminus v}$. Note that, by Lemma 10, the vertex $\overline{s}$ is uniquely defined for each $s \in V_{H''}$. Also see Figure 1 for an illustration.

3.3 A 2-Approximation Algorithm

Let $H'' = (V_{H''}, E_{H''})$ be the simplified implication graph we obtained from Section 3.2. Note that H'' is acyclic by Lemma 9. Now we are ready to describe our assignment LP LP-(T) for this problem and our 2-approximation algorithm.

For each vertex $s \in V_{H''}$, we introduce a decision variable $x_s \in \{0,1\}$ to indicate whether or not the assignments specified in the nodes of the SCC $\delta(s)$ should be realized. In this regard, for any $v \in V$, define $p_s(v) := \sum_{e \in E \text{ s.t. } u_{e,v} \in \delta(s)} p_{e,v}$ to be the workload vertex $v \in V$ will receive, if the assignments in $\delta(s)$ are realized.

Let $T \geq 0$ be the target maximum workload of the vertices to be achieved. We have the following feasibility LP relaxation with respect to the target value T.

$$\sum_{s \in V_{H''}} p_s(v) \cdot x_s + q_v \leq T, \qquad \forall v \in V, \tag{3a}$$

$$x_s + x_{\overline{s}} = 1, \qquad \forall s \in V_{H''}, \tag{3b}$$

$$x_s \leq x_{s'}, \qquad \forall (s,s') \in E_{H''}, \tag{3c}$$

$$x_s \geq 0, \qquad \forall s \in V_{H''}. \tag{3d}$$

In the above LP formulation, the constraint (3a) models the maximum workload T for each $v \in V$. The second constraint (3b) states that, for each conjugating pair of SCCs, exactly one type of orientation is made. The third constraint (3c) models the arc of implication in H'', namely, if $(s,s') \in E_{H''}$ and x_s is 1, then $x_{s'}$ must also be 1.

The Algorithm

Our algorithm goes as follows. First, it uses binary search to compute the smallest T_0 such that LP-(T_0) is feasible. Let $\hat{\sigma}$ be an optimal assignment for Ψ and $\hat{T}$ be the maximum workload of $\hat{\sigma}$. Then, it follows that T_0 must be a lower-bound of $\hat{T}$, since $\hat{\sigma}$ corresponds to a set of feasible solution for LP-($\hat{T}$). Let x^* be a fractional solution for LP-(T_0).

In the following, we describe a procedure that rounds x^* into an integer solution $\tilde{x}$ such that the workload of each vertex is at most doubled. Define

$$S^{\neq} := \left\{ s \in V_{H''} \;\middle|\; x_s^* \neq \frac{1}{2} \right\} \quad \text{and} \quad S^{=} := \left\{ s \in V_{H''} \;\middle|\; x_s^* = \frac{1}{2} \right\}.$$

For any $s \in S^{\neq}$, define

$$\tilde{x}_s := \begin{cases} 1, & \text{if } x_s^* > 1/2, \\ 0, & \text{if } x_s^* < 1/2. \end{cases}$$

By constraint (3b), if $x_s^* > 1/2$ for some $s \in S^{\neq}$, then it follows that $x_{\overline{s}}^* < 1/2$ and vice versa. Hence, the above setting of $\tilde{x}$ keeps constraint (3b) satisfied. Furthermore, the workload each vertex receives is at most doubled since x_s^* is rounded up only when it is at least $1/2$.

However, for any component $s \in S^{=}$, we have $\overline{s} \in S^{=}$ as well. Hence, x_s^* and $x_{\overline{s}}^*$ cannot both be rounded up at the same time since constraint (3b) will be violated. To resolve the rounding problem for components in $S^{=}$, we use the fact that $H'' \setminus S^{\neq}$ is still a DAG and consider the topological order of the components in $S^{=}$.

Let $S := H'' \setminus S^{\neq}$. Repeat the following steps until S becomes empty. In each iteration, pick a component $s \in S$ with zero out-degree. Intuitively, the orientation of s does not affect the orientation of the remaining components in S. We set $\tilde{x}_s$ to be 1 and $\tilde{x}_{\overline{s}}$ to be zero. Then we remove both s and $\overline{s}$ from S. This process is repeated until S becomes empty.

To obtain an orientation for the edges in E, we make the assignments specified in each SCC s with $\tilde{x}_s = 1$. In particular, for each $s \in V_{H''}$ with $\tilde{x}_s = 1$ and each node, say, $u_{e,v} \in \delta(s)$, we assign e to v by setting $\sigma(e) = v$. Then we output σ to be the approximate solution for Ψ.

The following lemma shows that, in any iteration of the above rounding procedure, if a component s has zero out-degree, then $\overline{s}$ must have a zero in-degree. This shows that our rounding procedure for components in $S^=$ is well-defined.

▶ **Lemma 11.** *For any $s \in V_{H''}$, if s has zero out-degree, then $\overline{s}$ must have a zero in-degree.*

The following lemma shows that σ is a feasible orientation for Ψ. Note that this also completes the proof for Lemma 5 and our characterization on the feasibility of $(2,2)$-SNC graph balancing.

▶ **Lemma 12.** *σ is feasible for Ψ.*

Proof. As an integer solution for LP-(T) corresponds naturally to a feasible assignment, it suffices to show that $\tilde{x}$ is feasible for LP-(T) for some T.

Clearly $\tilde{x}$ satisfies constraint (3b) and (3d) in LP-(T_0). For the constraint (3c), consider any $(s, s') \in E_{H''}$. Since x^* is a feasible solution for LP-(T_0), we have $x^*_s \le x^*_{s'}$. We will show that $\tilde{x}_s \le \tilde{x}_{s'}$. Depending on the values of x^*_s and $x^*_{s'}$, we consider the following cases.

- If $1/2 < x^*_s \le x^*_{s'}$ or $x^*_s \le x^*_{s'} < 1/2$, then $\tilde{x}_s = \tilde{x}_{s'}$ by our rounding scheme.
- If $x^*_s < 1/2$ or $1/2 < x^*_{s'}$, then $\tilde{x}_s = 0$ for the former case or $\tilde{x}_{s'} = 1$ for the latter case. In both cases, $\tilde{x}_s \le \tilde{x}_{s'}$ holds.
- For the remaining case for which $x^*_s = x^*_{s'} = 1/2$, assume for contradiction that constraint (3c) is not satisfied, i.e., $\tilde{x}_s = 1$ and $\tilde{x}_{s'} = 0$.
  Since $\tilde{x}_s = 1$, we know that s' has already been removed from H'' when s is selected to be rounded up by the algorithm. Since $\tilde{x}_{s'} = 0$, we know that s' was removed because its conjugating pair was selected and removed. But this will be a contradiction to Lemma 11 since the in-degree of s' was at least 1 at that time. Hence, constraint (3c) also holds.

This proves the feasibility of $\tilde{x}$ for LP-(T) for some T. ◀

It remains to prove the following theorem.

▶ **Theorem 13.** *σ can be computed in polynomial-time and is a 2-approximation for Ψ.*

Proof. It is clear that the computation can be done in polynomial-time. For each vertex $v \in V$, we know that the workload of v is

$$\sum_{e \in \sigma^{-1}(v)} p_{e,v} \;=\; \sum_{s \in V_{H''}} \tilde{x}_s \cdot \left(\sum_{u_{e,v} \in \delta(s)} p_{e,v} \right) \;+\; q_v \;=\; \sum_{s \in V_{H''}} \tilde{x}_s \, p_s(v) \;+\; q_v \;.$$

Observe that for any $s \in V_{H''}$, $\tilde{x}_s = 1$ only when $x^*_s \ge 1/2$. Hence we have $\tilde{x}_s \le 2 \cdot x^*_s$. It follows that, for each vertex $v \in V$, we have

$$\sum_{s \in V_{H''}} \tilde{x}_s \, p_s(v) \;+\; q_v \;\le\; \sum_{s \in V_{H''}} 2 \cdot x^*_s \, p_s(v) \;+\; q_v \;\le\; 2 \cdot T_0 \;\le\; 2 \cdot \hat{T},$$

where $\hat{T}$ is the maximum workload of the optimal assignment $\hat{\sigma}$ and in the last inequality we use the fact that T_0 is the smallest value such that LP-(T_0) is feasible. ◀

Integrality Gap of LP-(T)

In the following we show that the integrality gap of LP-(T) is 2. This shows that the approximation ratio we obtained for this problem is tight in terms of the LP we use. Consider the instance shown in Figure 2 with the weights $p_{e_1,a} = p_{e_5,b} = p_{e_3,a} = p_{e_3,b} = 1$ and all other 0, and the SNC constraints $\mathcal{C} = \{\{e_1, e_2\}, \{e_2, e_3\}, \{e_3, e_4\}, \{e_4, e_5\}\}$.

Observe that no matter e_3 is oriented to a or b, it always forces e_1 or e_5 to be oriented to the same vertex to which e_3 is oriented. Hence, the maximum workload of any feasible orientation is at least 2. On the other hand, consider the simplified implication graph H'' of the instance, shown on the r.h.s. of Figure 2. Observe that, by setting $x_s = 1/2$ for all $s \in V_{H''}$, all the constraints of LP-(T) with $T = 1$ are satisfied and we obtain a fractional orientation with maximum workload 1. This shows that the integrality gap is at least 2.

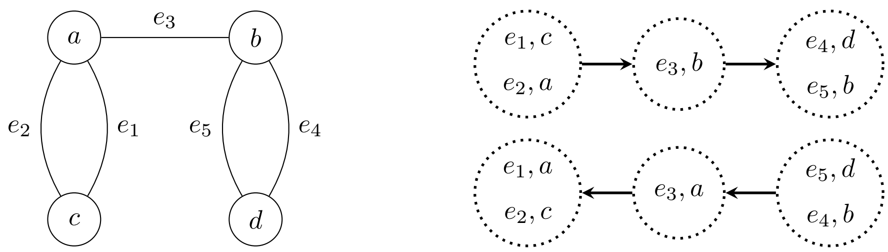

Figure 2 An example which shows that the integrality gap of LP-(T) is 2. On the right hand side, we use e, v to denote $u_{e,v}$ for notational simplicity.

Extension to Weighted Completion Time with SNC Constraints.

Our approach for graph balancing with SNC constraints is general and can be applied to similar settings to obtain similar approximation guarantees. In the following, we sketch how our algorithm framework can be used to obtain a 2-approximation when the objective is to minimize the weighted completion time, instead of maximum workload.

In fact, apart from the different objective function we need in the LP formulation, the remaining parts are exactly the same. We have the following corollary.

▶ **Corollary 14.** *We can compute a 2-approximation for the* $(2, 2)$*-SNC graph balancing problem to minimize the weighted completion time.*

4 Min-Max $(2, 2)$-SNC Graph Balancing on Restricted Graphs

In this section, we present both approximation and hardness results for min-max $(2, 2)$-SNC graph balancing on restricted graphs to describe a complete landscape of this problem.

Let $\Psi = (G = (V, E),\ p,\ \mathcal{C})$ be an instance of min-max $(2, 2)$-SNC graph balancing.

▶ **Theorem 15.** *There is an FPTAS for min-max* $(2, 2)$*-SNC graph balancing when* $|V| = 2$*. When* $|V| \geq 3$*, this problem is at least as hard as vertex cover to approximate.*

Note that, Theorem 15 provides a clear landscape on this problem and shows that, assuming the unique game conjecture (UGC), the approximation guarantee we obtained in this work is already tight even when $|V| = 3$.

References

1 Chidambaram Annamalai, Christos Kalaitzis, and Ola Svensson. Combinatorial algorithm for restricted max-min fair allocation. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1357–1372. SIAM, 2015. doi:10.1137/1.9781611973730.90.

2 Arash Asadpour, Uriel Feige, and Amin Saberi. Santa claus meets hypergraph matchings. In Ashish Goel, Klaus Jansen, José D. P. Rolim, and Ronitt Rubinfeld, editors, *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques, 11th International Workshop, APPROX 2008, and 12th International Workshop, RANDOM 2008, Boston, MA, USA, August 25-27, 2008. Proceedings*, volume 5171 of *Lecture Notes in Computer Science*, pages 10–20. Springer, 2008. doi:10.1007/978-3-540-85363-3_2.

3 Nikhil Bansal, Aravind Srinivasan, and Ola Svensson. Lift-and-round to improve weighted completion time on unrelated machines. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 156–167. ACM, 2016. doi:10.1145/2897518.2897572.

4 Nikhil Bansal and Maxim Sviridenko. The santa claus problem. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 31–40. ACM, 2006. doi:10.1145/1132516.1132522.

5 Alok Baveja, Xiaoran Qu, and Aravind Srinivasan. Approximating weighted completion time via stronger negative correlation. *Journal of Scheduling*, pages 1–10, 2023.

6 Ivona Bezáková and Varsha Dani. Allocating indivisible goods. *SIGecom Exch.*, 5(3):11–18, 2005. doi:10.1145/1120680.1120683.

7 Hans L. Bodlaender, Klaus Jansen, and Gerhard J. Woeginger. Scheduling with incompatible jobs. In Ernst W. Mayr, editor, *Graph-Theoretic Concepts in Computer Science, 18th International Workshop, WG '92, Wiesbaden-Naurod, Germany, June 19-20, 1992, Proceedings*, volume 657 of *Lecture Notes in Computer Science*, pages 37–49. Springer, 1992. doi:10.1007/3-540-56402-0_34.

8 Deeparnab Chakrabarty, Julia Chuzhoy, and Sanjeev Khanna. On allocating goods to maximize fairness. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, pages 107–116. IEEE Computer Society, 2009. doi:10.1109/FOCS.2009.51.

9 Deeparnab Chakrabarty, Sanjeev Khanna, and Shi Li. On (1,)-restricted assignment makespan minimization. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1087–1101. SIAM, 2015. doi:10.1137/1.9781611973730.73.

10 Deeparnab Chakrabarty and Kirankumar Shiragur. Graph balancing with two edge types. *CoRR*, abs/1604.06918, 2016. arXiv:1604.06918.

11 Siu-Wing Cheng and Yuchen Mao. Restricted max-min fair allocation. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 37:1–37:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.37.

12 Siu-Wing Cheng and Yuchen Mao. Restricted max-min allocation: Approximation and integrality gap. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 38:1–38:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.38.

13 Syamantak Das and Andreas Wiese. On minimizing the makespan when some jobs cannot be assigned on the same machine. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPIcs*, pages 31:1–31:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ESA.2017.31.

14 Sami Davies, Thomas Rothvoss, and Yihao Zhang. A tale of santa claus, hypergraphs and matroids. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 2748–2757. SIAM, 2020. doi:10.1137/1.9781611975994.167.

15 Irit Dinur, Oded Regev, and Clifford D. Smyth. The hardness of 3 - uniform hypergraph coloring. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, page 33. IEEE Computer Society, 2002. doi:10.1109/SFCS.2002.1181880.

16 Tomás Ebenlendr, Marek Krcál, and Jirí Sgall. Graph balancing: a special case of scheduling unrelated parallel machines. In Shang-Hua Teng, editor, *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 483–490. SIAM, 2008. URL: http://dl.acm.org/citation.cfm?id=1347082.1347135.

17 Uriel Feige. On allocations that maximize fairness. In Shang-Hua Teng, editor, *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 287–293. SIAM, 2008. URL: http://dl.acm.org/citation.cfm?id=1347082.1347114.

18 Penny Haxell and Tibor Szabó. Improved integrality gap in max-min allocation: or topology at the north pole. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 2875–2897. SIAM, 2023. doi:10.1137/1.9781611977554.ch109.

19 Chien-Chung Huang and Sebastian Ott. A combinatorial approximation algorithm for graph balancing with light hyper edges. In Piotr Sankowski and Christos D. Zaroliagis, editors, *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, volume 57 of *LIPIcs*, pages 49:1–49:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ESA.2016.49.

20 Klaus Jansen, Alexandra Lassota, and Marten Maack. Approximation algorithms for scheduling with class constraints. In Christian Scheideler and Michael Spear, editors, *SPAA '20: 32nd ACM Symposium on Parallelism in Algorithms and Architectures, Virtual Event, USA, July 15-17, 2020*, pages 349–357. ACM, 2020. doi:10.1145/3350755.3400247.

21 Klaus Jansen and Lars Rohwedder. On the configuration-lp of the restricted assignment problem. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2670–2678. SIAM, 2017. doi:10.1137/1.9781611974782.176.

22 Jan Karel Lenstra, David B. Shmoys, and Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 217–224. IEEE Computer Society, 1987. doi:10.1109/SFCS.1987.8.

23 Daniel R. Page and Roberto Solis-Oba. A 3/2-approximation algorithm for the graph balancing problem with two weights. *Algorithms*, 9(2):38, 2016. doi:10.3390/a9020038.

24 Daniel R. Page and Roberto Solis-Oba. Makespan minimization on unrelated parallel machines with a few bags. In Shaojie Tang, Ding-Zhu Du, David L. Woodruff, and Sergiy Butenko, editors, *Algorithmic Aspects in Information and Management - 12th International Conference, AAIM 2018, Dallas, TX, USA, December 3-4, 2018, Proceedings*, volume 11343 of *Lecture Notes in Computer Science*, pages 24–35. Springer, 2018. doi:10.1007/978-3-030-04618-7_3.

25 Tytus Pikies, Krzysztof Turowski, and Marek Kubale. Scheduling with complete multipartite incompatibility graph on parallel machines: Complexity and algorithms. *Artif. Intell.*, 309:103711, 2022. doi:10.1016/j.artint.2022.103711.

26 Petra Schuurman and Gerhard J Woeginger. Polynomial time approximation algorithms for machine scheduling: Ten open problems. *Journal of Scheduling*, 2(5):203–213, 1999.

27 Ola Svensson. Santa claus schedules jobs on unrelated machines. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 617–626. ACM, 2011. doi:10.1145/1993636.1993718.

28 José Verschae and Andreas Wiese. On the configuration-lp for scheduling on unrelated machines. In Camil Demetrescu and Magnús M. Halldórsson, editors, *Algorithms - ESA 2011 - 19th Annual European Symposium, Saarbrücken, Germany, September 5-9, 2011. Proceedings*, volume 6942 of *Lecture Notes in Computer Science*, pages 530–542. Springer, 2011. doi:10.1007/978-3-642-23719-5_45.

29 David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011. URL: http://www.cambridge.org/de/knowledge/isbn/item5759340/?site_locale=de_DE.