Breaking symmetries to rescue Sum of Squares: The case of makespan scheduling

Victor Verdugo†  José Verschae‡

Abstract

The Sum of Squares (SoS) hierarchy gives an automatized technique to create a family of increasingly tight convex relaxations for binary programs. There are several problems for which a constant number of rounds of the hierarchy give integrality gaps matching the best known approximation algorithm. In many other, however, ad-hoc techniques give significantly better approximation ratios. Notably, the lower bounds instances, in many cases, are invariant under the action of a large permutation group. The main purpose of this paper is to study how the presence of symmetries on a formulation degrades the performance of the relaxation obtained by the SoS hierarchy. We do so for the special case of the minimum makespan problem on identical machines. Our first result is to show that a linear number of rounds of SoS applied over the configuration linear program yields an integrality gap of at least 1.0009. This improves on the recent work by Kurpisz et al. [40] that shows an analogous result for the weaker LS+ and SA hierarchies. Then, we consider the weaker assignment linear program and add a well chosen set of symmetry breaking inequalities that removes a subset of the machine permutation symmetries. We show that applying the SoS hierarchy for \( O(1) \) rounds to this linear program reduces the integrality gap to \( (1 + \varepsilon) \). Our results suggest that for this classical problem the symmetries of the natural assignment linear program were the main barrier preventing the SoS hierarchy to give relaxations with integrality gap \( (1 + \varepsilon) \) after a constant number of rounds. We leave as an open question whether this phenomenon occurs for different problems where the SoS hierarchy yields weak relaxations.

*This work has been partially funded by Project Fondecyt Nr. 1181527.
†Institute of Engineering Sciences, Universidad de O’Higgins. victor.verdugo@uoh.cl
‡Institute of Engineering Sciences, Universidad de O’Higgins. jose.verschae@uoh.cl


## Contents

1 Introduction .................................................. 2
  1.1 Related work .............................................. 4

2 Preliminaries: Sum of Squares (SoS) and Pseudoexpectations ............ 5

3 Lower bound: Symmetries are hard for SoS ................................ 7
  3.1 Configuration Linear Program ................................ 7
  3.2 A symmetry-reduced decomposition of the scheduling ideal ............. 8
  3.3 Spanning sets of the scheduling ideal ................................ 11
  3.4 Spanning sets of the invariant row subspace .......................... 12
  3.5 High-degree SoS pseudoexpectation: Proof of Theorem 1 ............... 14

4 Upper bound: Breaking symmetries to approximate with SoS ............... 20
  4.1 Assignment Linear Program ...................................... 20
  4.2 Symmetry breaking inequalities .................................... 21
  4.3 Balanced partitionings ............................................ 24
  4.4 SDP based approximation scheme: Proof of Theorem 2 .................. 25
  4.5 Pseudoexpectation rounding ....................................... 26

5 Appendix ................................................................ 35

## 1 Introduction

Lift-and-project methods are powerful techniques for deriving convex relaxations of integer programs. The lift-and-project hierarchies, as Sherali & Adams (SA), Lovász & Schrijver (LS), or Sum of Squares (SoS), are systematic methods for obtaining a family of increasingly tight relaxations, parameterized on the round of the hierarchy. For all this hierarchies, r rounds on a problem with n variables imply a convex relaxation with \( n^{O(r)} \) variables in the lifted space. Taking \( r = n \) rounds, gives an exact description of the integer hull, at the cost of having an exponential number of variables. Arguably, it is not well understood for which problems these hierarchies yield relaxations that match the best possible approximation algorithm. Indeed, there are some positive results, but many other strong negative results for algorithmically easy problems. This shows a natural limitation to the power of hierarchies as one-fit-all techniques. Quite remarkably, the instances used for obtaining lower bounds often have a very symmetric structure [43, 21, 56, 40, 58], which suggests a strong connection between the tightness of the relaxation given by these hierarchies and symmetries. The main purpose of this article is to study this connection for a specific relevant problem, namely, minimum makespan scheduling on identical machines.

**Minimum makespan scheduling.** This problem is one of the first problems considered under the lens of approximation algorithms [20], and since then it has been studied extensively. The input of the problem consists of a set \( J \) of \( n \) jobs, each having an integral processing time \( p_j > 0 \), and a set \( M = [m] \) of \( m \) identical machines. Given an assignment \( \sigma : J \to M \), the load of a machine \( i \) is the total processing time of jobs assigned to \( i \), that is, \( \sum_{j \in \sigma^{-1}(i)} p_j \). The objective is to find an assignment of jobs to machines that minimizes the makespan, that is, the maximum load. The problem is
strongly NP-hard and admits several polynomial-time approximation schemes (PTASs) based on different techniques, as dynamic programming, integer programming on fixed dimension, and integer programming under a constant number of constraints [26, 1, 2, 25, 29, 30, 15].

**Integrality gaps.** The minimum makespan problem has two natural linear relaxations which have been extensively studied in the literature. The assignment linear program uses binary variables $x_{ij}$ which indicate whether job $j$ is assigned to machine $i$. It is easy to see that its integrality gap is 2. The stronger configuration linear program, uses an exponential number of variables $y_C$ which indicate whether the set of jobs assigned to $i$ has $C$ as a multiset of processing times. Kurpisz et al. [40] showed that the configuration linear program has an integrality gap of at least $1024/1023 \approx 1.0009$ even after a linear number of rounds of the LS$_+$ or SA hierarchies. Hence, the same lower bound holds when the ground formulation is the assignment linear program. On the other hand, Kurpisz et al. [40] leave open whether the SoS hierarchy applied to the configuration linear program has a $(1 + \varepsilon)$ integrality gap after $O(1)$ many rounds. Our first main contribution is a negative answer to this question.

**Theorem 1.** Consider the problem of scheduling identical machines to minimize the makespan. For each $n \in \mathbb{N}$ there exists an instance with $n$ jobs such that, after applying $\Omega(n)$ rounds of the SoS hierarchy over the configuration linear program, the obtained semidefinite relaxation has an integrality gap of at least 1.0009.

Naturally, since the configuration linear program is stronger than the assignment linear program, our result holds if we apply $\Omega(n)$ rounds of SoS over the assignment linear program. The proof of the lower bound relies on tools from representation theory of symmetric groups over polynomials rings and it is inspired on the recent work by Raymond et al. for symmetric sums of squares in hypercubes [59]. The lower bound comes by constructing high-degree pseudoexpectations on one hand, and by obtaining symmetry-reduced decompositions of the polynomial ideal defined by the configuration linear program, on the other hand. The machinery from representation theory allows to restrict attention to invariant polynomials, and we combine this with a strong pseudoindependency result for a well chosen polynomial spanning set. Our analysis is also connected to the work of Razborov on flag algebras and graph densities, and we believe it can be of independent interest for analyzing lower bounds in the context of SoS in presence of symmetries [60, 61, 58].

**Symmetries and Hierarchies.** It is natural to explore whether symmetry handling techniques might help overcoming the limitation given by Theorem 1. A natural source of symmetry the problem comes from the fact that the machines are identical: Given a schedule, we obtain other with the same makespan by permuting the assignment over the machines. The same symmetries are encountered in the assignment and configuration linear programs, namely, if $\sigma : M \to M$ is a permutation and $(x_{ij})$ is a feasible solution to the assignment linear program then $(x_{\sigma(i)j})$ is also feasible. In other words, the assignment linear program is invariant under the action of the symmetric group on the set of machines. The question we study is the following: Is it possible to obtain a polynomial size linear or semidefinite program with an integrality gap of at most $(1 + \varepsilon)$ that is not invariant for the machine symmetries? That is, our goal is to understand if the group action is deteriorating the quality of the relaxations obtained from the SoS hierarchy. This time, we provide a positive answer.

**Theorem 2.** Consider the problem of scheduling identical machines to minimize the makespan. After adding a set of linearly many symmetry breaking inequalities to the assignment linear program, $O(\varepsilon(1))$ rounds of the SoS hierarchy yields a convex relaxation with an integrality gap of at most $(1 + \varepsilon)$, for any $\varepsilon > 0$. 

3
The theorem is based on introducing a formulation that breaks the symmetries in the invariant assignment program by adding new constraints. This enforces that any integer feasible solution of the formulation should respect a lexicographic order over the machine configurations. On top of the linear program obtained from adding the aforementioned constraints, we apply the SoS hierarchy. Using the decomposition theorem [33], we can construct a solution that is integral on a well chosen set of machines $M'$ of size $O_\varepsilon(1)$. Our symmetry breaking inequalities imply that between two consecutive machines in $M'$, our solution assigns approximately the same configurations, and thus we can construct an approximately optimal solution.

![Figure 1: The polytope in $\mathbb{R}^2$ at the left is invariant under the action of permuting variables, that is, every time $(x_1, x_2)$ is feasible, then $(x_2, x_1)$ is also feasible.](image)

### 1.1 Related work

**Upper bounds.** The first application of semidefinite programming in the context of approximation algorithms was by the work of Goemans & Williamson for Max-Cut [19]. There are not many positive results in this line for other combinatorial optimization problems, but of particular interest to our work is the SoS based approximation scheme by Karlin et al. to the Max-Knapsack problem [33]. They use a structural decomposition result satisfied by the SoS hierarchy, and which makes a difference respect to other classic hierarchies. Recently, for a constant number of machines, Levey and Rothvoss design an approximation scheme with a sub-exponential number of rounds in the weaker SA hierarchy [45]. A lot of attention has received the SoS method in order to design algorithms for high-dimensional problems. Among them we find matrix and tensor completion [8, 57], tensor decomposition [50] and clustering [36]. See the recent survey of Raghavendra et al. for high-dimensional estimation using the SoS method [58]. In the context of hierarchies we refer to Laurent [42] for a detailed comparison between SoS and others. For applications in approximation algorithms we refer to the survey of Rothvoss [63].

**Lower bounds.** The first was obtained in the context of positivstellensatz certificates was by Grigoriev [21], showing the necessity of a linear number of SoS rounds to refute an easy Knapsack instance. A similar result was obtained by Laurent [43] on the number of rounds needed to certificate the infeasibility of certain Max-Cut instances, and recently the work of Kurpisz et al. in unconstrained polynomial optimization [37]. The same authors show that for a certain polynomial-time single machine scheduling problem, the SoS hierarchy exhibits an unbounded integrality gap even
in high-degree regime [37, 39]. Remarkable are the work of Grigoriev [22] and Schoenebeck [65] exhibiting the difficulty for SoS to certify the insatisfiability of a family of random 3-SAT instances in subexponential time, and recently there have been efforts on unifying frameworks to show lower bounds on random CSP’s [6, 35, 34]. For estimation and detection problems, lower bounds have been shown for the \( k \)-clique problem, \( k \)-densest subgraph and tensor PCA, among others [27, 7].

**Invariant Sum of Squares.** Remarkable in this line is the work of Gatermann & Parrillo, that studied how to obtain reduced sums of squares certificates of non-negativity when the polynomial is invariant under the action of a group, using tools from representation theory [16]. Recently, Raymond et al. developed on the Gatermann & Parrillo method to construct symmetry-reduced sum of squares certificates for polynomials over \( k \)-subset hypercubes [59]. Furthermore, the authors make an interesting connection with the Razborov method and flag algebras [60, 61]. Blekherman et al. provided degree bounds on rational representations for certificates over the hypercube, recovering as corollary known lower bounds for combinatorial optimization problems like Max-Cut [10, 44]. Other applications of symmetries in semidefinite and linear programming can be found in combinatorial optimization [62, 5, 24, 38, 14], graph theory [23, 13, 12, 56] and coding theory [17, 4, 48].

**Symmetry Handling in Integer Programming.** The integer programming community have dealt with symmetries by either breaking them [32, 46, 28], or devising symmetry-aware exact algorithms as isomorphism pruning [51], orbital branching [54] and orbitopal fixing [31]. The work by Ostrowski [53] combines hierarchies and symmetry handling but with a fundamentally different approach as ours. The author uses the SA hierarchy and reduces the dimension of the lifted relaxation to obtain a faster algorithm. It is worth noticing that such approach does not help reducing the exponential dependency on the number of rounds nor helps diminishing the integrality gap. For an extensive treatment we refer to the surveys by Margot [52] and Liberti [47].

## 2 Preliminaries: Sum of Squares (SoS) and Pseudoexpectations

In what follows we denote by \( \mathbb{R}[x] \) the ring of polynomials with real coefficients. Binary integer programming belongs to a larger class of problems in polynomial optimization, where the constraints are defined by polynomials in the variables indeterminates. More specifically, consider the set

\[
\mathcal{K} = \left\{ x \in \mathbb{R}^E : g_i(x) \geq 0 \text{ for all } i \in M, \ h_j(x) = 0 \text{ for all } j \in J, \ x_e^2 - x_e = 0 \text{ for all } e \in E \right\}, \tag{1}
\]

where \( g_i, h_j \in \mathbb{R}[x] \) for all \( i \in M \) and for all \( j \in J \). In particular, for binary integer programming the equality and inequality constraints are affine functions.

**Ideals, quotients and square-free polynomials.** We denote by \( \mathbf{I}_E \) the ideal of polynomials in \( \mathbb{R}[x] \) generated by \( \{ x_e^2 - x_e : e \in E \} \), and let \( \mathbb{R}[x]/\mathbf{I}_E \) be the quotient ring of polynomials that vanish in the ideal \( \mathbf{I}_E \). That is, \( f, g \in \mathbb{R}[x] \) are in the same equivalence class of the quotient ring if \( f - g \in \mathbf{I}_E \), that we denote \( f \equiv g \mod \mathbf{I}_E \). Alternatively, \( f \equiv g \mod \mathbf{I}_E \) if and only if the polynomials evaluate to the same values on the vertices of the hypercube, that is, \( f(x) = g(x) \) for all \( x \in \{0, 1\}^E \).

**Example 1.** The polynomial \( f = 3x_1^2x_2 + 7x_1x_2^2 - 10x_1x_2 \) is in the ideal generated by the polynomials \( x_1^2 - x_1 \) and \( x_2^2 - x_2 \), since

\[
3x_1^2x_2 + 7x_1x_2^2 - 10x_1x_2 = 3x_2(x_1^2 - x_1) + 7x_1(x_2^2 - x_2),
\]

(5)
The question of certifying the infeasibility of $\sum_{\alpha \in \mathcal{A}} s_{\alpha}^{\ell}$ is in the quotient ring such that $f \equiv \sum_{\alpha \in \mathcal{A}} s_{\alpha}^{\ell} \mod I_E$.

Certificates and SoS method. The question of certifying the infeasibility of (1) is hard in general but sometimes it is possible to find simple certificates of infeasibility. We say that there exists a degree-$\ell$ SoS certificate of infeasibility for $K$ if there exist SoS polynomials $s_0$ and $\{s_i\}_{i \in M}$, and polynomials $\{r_j\}_{j \in J}$, all of them in the quotient ring, such that

$$-1 \equiv s_0 + \sum_{i \in M} s_i g_i + \sum_{j \in J} r_j h_j \mod I_E, \quad (2)$$

and the degree of every polynomial in the right hand side is at most $\ell$. Observe that if $K$ is feasible, then the right hand side is guaranteed to be non negative for at least one assignment of $x$ in $\{0, 1\}^E$, which contradicts the equality above. In the case of binary integer programming, if $K$ is infeasible there exists a degree-$\ell$ SoS certificate, with $\ell \leq |E|$ [42, 55].

Example 2. Given $\varepsilon > 0$, consider the program $x_1 + x_2 = 1$ and $x_1 x_2 - \varepsilon \geq 0$, with $x_1, x_2 \in \{0, 1\}$. This program is infeasible since the equality constraint forces that exacty one of the variables is one and the other is zero, that is, their product is null. Let $s_0 = 0$, $s_1 = 1/\varepsilon$ and $r = -(x_1 + x_2)/2\varepsilon$. We check that they provide a degree-2 SoS certificate of infeasibility,

$$\frac{1}{\varepsilon} (x_1 x_2 - \varepsilon) - \frac{1}{2\varepsilon} (x_1 + x_2)(x_1 + x_2 - 1) = \frac{1}{\varepsilon} (x_1 x_2 - \varepsilon) - \frac{1}{2\varepsilon} \cdot 2 x_1 x_2 \equiv -1 \mod I_{\{1,2\}}.$$

The SoS algorithm iteratively checks the existence of a SoS certificate, parameterized in the degree, and each step of the algorithm is called a round. Since $|E|$ is an upper bound on the certificate degree, the method is guaranteed to finish [55, 9]. Furthermore, the existence of a degree-$\ell$ SoS certificate can be decided by solving a semidefinite program, in time $|E|^{O(\ell)}$. This approach can be seen as the dual of the hierarchy proposed by Lasserre, which has been studied extensively in the optimization and algorithms community [41, 42, 63, 11].

Pseudoexpectations. To determine the existence of a SoS certificate one solves a semidefinite program, and the solutions of this programs determine the coefficients of elements in the dual space of linear operators. We say that a linear functional $\tilde{E} : \mathbb{R}[x]/I_E \to \mathbb{R}$ is a degree-$\ell$ SoS pseudoexpectation for (1), if it satisfies the following properties:

1. $\tilde{E}(1) = 1$,
2. $\tilde{E}(f^2) \geq 0$ for all $f \in \mathbb{R}[x]/I_E$ with $\deg(f) \leq \ell/2$,
3. $\tilde{E}(f^2 g_i) \geq 0$ for all $i \in M$, for all $f \in \mathbb{R}[x]/I_E$ with $\deg(f^2 g_i) \leq \ell$,
Lemma 1. Suppose that $K$ defined in (1) is infeasible. If there exists a degree-$\ell$ SoS pseudoexpectation for $K$ then there is no degree-$\ell$ SoS certificate of infeasibility.

Proof. Suppose there exists a degree-$\ell$ certificate of infeasibility for $K$, that is, $s_0$ and $\{s_i\}_{i \in M}$ SoS polynomials, and $\{r_j\}_{j \in J}$ satisfying (2), and let $\tilde{E}$ the degree-$\ell$ pseudoexpectation. Property (1) and linearity of the pseudoexpectation implies that $\tilde{E}(-1) = -1$, and

$$
\tilde{E} \left( s_0 + \sum_{i \in M} s_i g_i + \sum_{j \in J} r_j h_j \right) = \tilde{E}(s_0) + \sum_{i \in M} \tilde{E}(s_i g_i) + \sum_{j \in J} \tilde{E}(r_j h_j) \geq 0,
$$
due to linearity and properties (2)-(4) of the pseudoexpectation. This yields a contradiction. 

The minimum value of $\ell$ for which there exists a SoS certificate of infeasibility tells how hard is the program (1) for the SoS method. Lemma 1 provides a way of finding lower bounds on the minimum value of certificate degree, and we use it later for studying this number in the context of scheduling, in Section 3. The higher the degree of a pseudoexpectation, the higher is the minimum degree of certificate of infeasibility. There are many examples of problems that are extremely easy to certificate for humans, but not for the SoS method. For example, given a positive $k \in \mathbb{Q} \setminus \mathbb{Z}$, consider the program $\sum_{e \in E} x_e = k$ and $x_e^2 - x_e = 0$ for all $e \in E$. This problem is clearly infeasible, but there is no degree-$\ell$ SoS certificate of infeasibility for $\ell \leq \min \{2 \lfloor k \rfloor + 3, 2 \lfloor n - k \rfloor + 3, n \}$, as shown originally by Grigoriev and others recently using different approaches [21, 56]. In the following we refer to low-degree when the degree of a certificate or the pseudoexpectation is $O(1)$.

Sherali & Adams certificates. There is a weaker notion of certificates obtained using linear programming due to Sherali & Adams (SA) [66]. In the case of equality constrained programs they correspond to find linear operators satisfying properties (1) and (4), and we say they are a degree-$\ell$ SA pseudoexpectation.

3 Lower bound: Symmetries are hard for SoS

In this section we show that the SoS method fails to provide a low-degree certificate of infeasibility for a certain family of scheduling instances. The program we analyze in this section is known as the configuration linear program, that has proven to be powerful for different scheduling and packing problems [67, 18].

3.1 Configuration Linear Program

Given a value $T > 0$, a configuration corresponds to a multiset of processing times such that its total sum does not exceed $T$. The multiplicity $m(p, C)$ indicates the number of times that the processing time $p$ appears in the multiset $C$. The load of a configuration $C$ is just the total processing time, $\sum_{p \in \{p_j \mid j \in J\}} m(p, C) \cdot p$. Given $T$, let $C$ denote the set of all configurations with load at most $T$. 

4. $\tilde{E}(fh_j) = 0$ for all $j \in J$, for all $f \in \mathbb{R}[x]/I_E$ with $\deg(fh_j) \leq \ell$. 

For each combination of a machine \(i \in M\) and a configuration \(C \in C\), the program has a variable \(y_{iC}\) that models whether machine \(i\) is scheduled with jobs with processing times according to configuration \(C\). Letting \(n_p\) denote the number of jobs in \(J\) with processing time \(p\), we can write the following binary linear program, \(\text{clp}(T)\),

\[
\sum_{C \in C} y_{iC} = 1 \quad \text{for all } i \in M,
\]

\[
\sum_{i \in M} \sum_{C \in C} m(p, C) y_{iC} = n_p \quad \text{for all } p \in \{p_j : j \in J\},
\]

\[
y_{iC} \in \{0, 1\} \quad \text{for all } i \in M, \text{ all } C \in C.
\]

**Hard instances.** We briefly describe the construction of a family of hard instances \(\{I_k\}_{k \in \mathbb{N}}\) for the configuration linear program in [40]. Let \(T = 1023\), and for each odd \(k \in \mathbb{N}\) we have \(n = 15k\) jobs and \(3k\) machines. There are 15 different job-sizes with value \(O(1)\), each one with multiplicity \(k\). There exist a set of special configurations \(\{C_1, \ldots, C_6\}\), called matching configurations, such that the program above is feasible if and only if the program restricted to the matching configurations is feasible. The infeasibility of the latter program comes from the fact that there is no 1-factorization of a regular multigraph version of the Petersen graph [40, Lemma 2].

**Theorem 3 ([40]).** For each odd \(k \in \mathbb{N}\), there exists a degree-\(\lfloor k/2 \rfloor\) SA pseudoexpectation for the configuration linear program. In particular, there is no low-degree SA certificate of infeasibility.

### 3.2 A symmetry-reduced decomposition of the scheduling ideal

In what follows, we consider the set of machines \(M = [m]\). Given \(T > 0\), the variables ground set for configuration linear program is \(E = [m] \times C\), and the symmetric group \(S_m\) acts over the monomials in \(\mathbb{R}[y]\) according to \(\sigma y_{iC} = y_{\sigma(i)C}\), for every \(\sigma \in S_m\). The action extends linearly to \(\mathbb{R}[y]/I_E\), and the configuration linear program is invariant under this action, that is, for every \(y \in \text{clp}(T)\) and every \(\sigma \in S_m\) we have \(\sigma y \in \text{clp}(T)\). We say that a polynomial \(f \in \mathbb{R}[y]/I_E\) is \(S_m\)-invariant if \(\sigma f = f\) for every \(\sigma \in S_m\). In particular, if \(f\) is invariant we have \(f = 1/|S_m| \sum_{\sigma \in S_m} \sigma f := \text{sym}(f)\), which is the symmetrization or Reynolds operator of the group action.
We say that a linear function $L$ over the quotient ring is $S_m$-symmetric if for every polynomial $f \in \mathbb{R}[y]/I_E$ we have $L(f) = L(\text{sym}(f))$. For the rest of this section we restrict attention to programs defined by equality constraints, as it is the case for the configuration linear program.

Lemma 2. Let $\tilde{E}$ be a symmetric linear operator over $\mathbb{R}[y]/I_E$ such that for every invariant SoS polynomial $g$ of degree at most $\ell$ we have $\tilde{E}(g) \geq 0$. Then, $\tilde{E}(f^2) \geq 0$ for every $f \in \mathbb{R}[y]/I_E$ with $\deg(f) \leq \ell/2$.

That is, when $\tilde{E}$ is symmetric it is enough to check the condition in the lemma above to satisfy (2). Therefore, in this case we restrict our attention to those polynomials that are invariant and SoS.

Proof of Lemma 2. Since the operator $\tilde{E}$ is symmetric, for every $f$ in the quotient ring with $\deg(f) \leq \ell/2$ we have $\tilde{E}(f^2) = \tilde{E}(\text{sym}(f^2))$. The polynomial $\text{sym}(f^2)$ is symmetric, and it is SoS since $\text{sym}(f^2) = 1/|S_m| \sum_{\sigma \in S_m} \sigma f^2$, which is a sum of squares. Since $\deg(\text{sym}(f^2)) \leq \ell$, we have $\tilde{E}(\text{sym}(f^2)) \geq 0$ and we conclude that $\tilde{E}(f^2) \geq 0$.

In the following we focus on understanding polynomials that are invariant and SoS. To analyze the action of the symmetric group over $\mathbb{R}[y]$ we introduce some tools from representation theory [64] to characterize the invariant $S_m$-modules of the polynomial ring. We maintain the exposition minimally enough for our purposes and we follow in part the notation used by Raymond et al. [59]. We refer to [64] for a deeper treatment of representation theory of symmetric groups.

Isotypic decompositions. We say that a $S_m$-module $V$ is irreducible if the only invariant subspaces are $\{0\}$ and $V$. Any $S_m$-module $V$ can be decomposed into irreducible modules, and the decomposition is indexed by the partitions of $m$. A partition of $m$ is a vector $(\lambda_1, \ldots, \lambda_t)$ such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_t > 0$ and $\lambda_1 + \cdots + \lambda_t = m$. We denote by $\lambda \vdash m$ when $\lambda$ is a partition of $m$. Then, $V$ can be decomposed as

$$V = \bigoplus_{\lambda \vdash m} V_\lambda,$$

that is, a direct sum where each $V_\lambda$ is an irreducible $S_m$-module of $V$ [64]. Each of the subspaces in the direct sum is called an isotypic component. A tableau of shape $\lambda$ is a bijective filling between $[m]$ and the cells of a grid with $t$ rows, and every row $r \in [t]$ has length $\lambda_r$. In this case, the shape or Young diagram of the tableau is $\lambda$. For a tableau $\tau_\lambda$ of shape $\lambda$, we denote by row $(\tau_\lambda)$ the subset of $[m]$ that fills row $r$ in the tableau.

Example 3. Let $m = 7$ and consider the partition $\lambda = (4, 2, 1)$. The following tableaux have shape $\lambda$,

---

1Think as $V = \mathbb{R}[y]$ to fix ideas.
In the tableau $\tau_\lambda$ at the left, row 1($\tau_\lambda$) = \{1, 2, 3, 4\}. In the tableau $\sigma_\lambda$ at the right, row 3($\sigma_\lambda$) = \{4\}.

The row group $R_{\tau_\lambda}$ is the subgroup of $S_m$ that stabilizes the rows of the tableau $\tau_\lambda$, that is,

$$R_{\tau_\lambda} = \left\{ \sigma \in S_m : \sigma \cdot \text{row}_r(\tau_\lambda) = \text{row}_r(\tau_\lambda) \text{ for every } r \in [t] \right\}. \quad (4)$$

**Invariant SoS polynomials.** We go back now to the case of the configuration linear program. Let $Q^\ell$ be the quotient ring $\mathbb{R}[y]/I_E$ restricted to polynomials of degree at most $\ell$ and let $Q^\ell = \bigoplus_{\lambda \vdash m} Q^\ell_\lambda$ be its isotypic decomposition. Given a tableau $\tau_\lambda$ of shape $\lambda$, let $W_{\tau_\lambda}$ the row subspace of fixed points in $Q^\ell$ for the row group $R_{\tau_\lambda}$, that is,

$$W_{\tau_\lambda} = \left\{ q \in Q_\lambda^\ell : \sigma q = q \text{ for all } \sigma \in R_{\tau_\lambda} \right\}. \quad (5)$$

It can be shown that for any tableau $\tau_\lambda$ of shape $\lambda$, the dimension $\dim(W_{\tau_\lambda})$ is the same value $m_\lambda$ [59, Lemma A.10]. The following result follows from the work of Gatermann & Parrilo in the context of symmetry reduction for invariant semidefinite programs [16]. They use it to show that an invariant semidefinite program can be decomposed into many programs of smaller dimension, one per isotypic module. In what follows, $(A, B)$ is the inner product in the space of square matrices defined by the trace of $AB$. Given $\ell \in [m]$, we denote by $\Lambda_\ell$ the subset of partitions of $m$ that are lexicographically larger than $(m - \ell, 1^\ell)$.

**Theorem 4.** Suppose that $g \in \mathbb{R}[y]/I_E$ is a degree-$\ell$ SoS and $S_m$-invariant polynomial. For each partition $\lambda \in \Lambda_\ell$, let $\tau_\lambda$ be a tableau of shape $\lambda$ and let $P^\lambda = \{p_1^\lambda, \ldots, p_r^\lambda\}$ be a set of polynomials such that $\text{span}(P^\lambda) \supseteq W_{\tau_\lambda}$. Then, for each partition $\lambda \in \Lambda_\ell$ there exists a $\ell \times \ell$ positive semidefinite matrix $M_\lambda$ such that $g = \sum_{\lambda \in \Lambda_\ell} \langle M_\lambda, Z^\lambda \rangle$, where $Z_{ij}^{\lambda} = \text{sym}(p_i^\lambda p_j^\lambda)$.

**Remark 1.** The theorem above is based on the recent work of Raymond et al. [59, p. 324, Theorem 3]. The key facts is that the number of partitions needed in the decomposition is reduced to a number that does not depend on $m$, and that it is enough to have a spanning set for each isotypic module, which is a relaxation from the original result of Gaterman & Parrilo that required a basis [16]. In our case the symmetric group is acting differently from Raymond et al., but the proof follows the same lines. We include a proof of our version in the Appendix for completeness.

Together with Lemma 2, it is enough to study pseudoexpectations for each of the partitions in $\Lambda_\ell$ separately. In particular, Theorem 4 gives us flexibility in the spanning set that we use for describing the row subspaces. We remark that for each partition we can take any tableau with that shape, and consider a spanning set for its corresponding row subspace. In the following, for a matrix $A$ with entries in $\mathbb{R}[y]/I_E$, let $\overline{E}(A)$ be the matrix obtained by applying $\overline{E}$ to each entry of $A$.

**Lemma 3.** Suppose that for each partition $\lambda \in \Lambda_\ell$, the spanning set $P^\lambda$ of $W_{\tau_\lambda}$ is such that $\overline{E}(Z^\lambda)$ is positive semidefinite. Then, for each $f$ with $\deg(f) \leq \ell/2$ we have $\overline{E}(f^2) \geq 0$. 

1 2 7 4

3 6

5 6

1 7 2 5

3 4

In the tableau $\tau_\lambda$ at the left, row 1($\tau_\lambda$) = \{1, 2, 3, 4\}. In the tableau $\sigma_\lambda$ at the right, row 3($\sigma_\lambda$) = \{4\}.
Proof. By Lemma 2 it is enough to prove the claim for \( g \) invariant and degree-\( \ell \) SoS. By Theorem 4, for each \( \lambda \in \Lambda_\ell \) there exist a positive semidefinite matrix \( M_\lambda \) such that \( g = \sum_{\lambda \in \Lambda_\ell} \langle M_\lambda, Z^\lambda \rangle \). Therefore, \( \tilde{\mathbb{E}}(g) = \tilde{\mathbb{E}} \left( \sum_{\lambda \in \Lambda_\ell} \langle M_\lambda, Z^\lambda \rangle \right) = \sum_{\lambda \in \Lambda_\ell} \langle M_\lambda, \tilde{\mathbb{E}}(Z^\lambda) \rangle \geq 0 \), since both \( M_\lambda \) and \( \tilde{\mathbb{E}}(Z^\lambda) \) are positive semidefinite for each partition \( \lambda \in \Lambda_\ell \).

3.3 Spanning sets of the scheduling ideal

In this section we show how to construct the spanning sets of the row subspaces in order to apply Lemma 3, which together with a particular linear operator provides the existence of a high-degree SoS pseudoexpectation. The structure of the configuration linear program allows us to further restrict the canonical spanning set obtained from monomials, by one that is combinatorially interpretable and adapted to our purposes.

Partial schedules. We say that \( S \subseteq [m] \times \mathbb{C} \) is a partial schedule if for every \( i \in [m] \) we have \( \delta_S(i) \leq 1 \), where \( \delta_S \) is the vertex degree in the (directed) bipartite graph \( G_S \) with vertex partition \([m]\) and \( \mathcal{C} \), and edges \( S \). For convenience, we say that \( S \) is a partial schedule over \( H \) if \( \{i : (i, C) \in S\} \subseteq H \). We denote by \( \mathcal{M}(S) \) the set of machines incident to a partial schedule \( S \), that is, \( \{i \in [m] : \delta_S(i) = 1\} \).

Sometimes is convenient to see a partial schedule \( S \) as a function from \( \mathcal{M}(S) \) to \( \mathcal{C} \), so we also say that \( S \) is partial schedule with domain \( \mathcal{M}(S) \).

Example 4. Suppose that \( m = 4 \) and \( \mathcal{C} = \{C_1, C_2, C_3\} \). The set \( T = \{(1, C_1), (2, C_1), (4, C_2)\} \) is a partial schedule. The machine \( i = 3 \) is not incident to \( T \). In this case, \( \delta_T(C_1) = 2 \) since there are two machines, \( \{1, 2\} \), incident to \( C_1 \). The domain of \( T \) is \( \mathcal{M}(T) = \{1, 2, 4\} \). The set \( S = \{(1, C_1), (1, C_2)\} \) is not a partial schedule since \( \delta_S(1) = 2 \).

Scheduling ideal. Let \( \text{sched} \) be the ideal of polynomials in \( \mathbb{R}[y]/I_E \) generated by

\[
\left\{ \sum_{C \in \mathcal{C}} y_i C - 1 : i \in [m] \right\} \cup \left\{ y_{iC}^2 - y_{iC} : i \in [m], C \in \mathcal{C} \right\},
\]

(6)

Recall that the set of polynomials above enforce the machines in the scheduling solutions to be assigned with exactly one configuration. In the following lemmas we show that this set of constraints induce a nice structure for constructing spanning sets in the quotient ring.

Lemma 4. If \( S \) is not a partial schedule, \( y_S \equiv 0 \mod \text{sched} \).

Proof. Since \( S \) it is not a partial schedule, there exists a machine \( i \in [m] \) and two configurations \( C, \tilde{C} \in \mathcal{C} \) such that \( (i, C_1), (i, C_2) \in S \). Then it is enough to prove that for every pair of different configurations \( C_1, C_2 \) we have \( y_{iC_1} y_{iC_2} \equiv 0 \mod \text{sched} \). To that end, fix configuration \( C_1 \) and we have

\[
\sum_{C \in \mathcal{C}, C \neq C_1} y_{iC_1} y_{iC} \equiv \sum_{C \in \mathcal{C}, C \neq C_1} y_{iC_1} y_{iC} + y_{iC_1}^2 - y_{iC_1} \equiv y_{iC_1} \left( \sum_{C \in \mathcal{C}} y_{iC} - 1 \right) \equiv 0 \mod \text{sched},
\]

and then we conclude the claim. In particular, \( y_S \equiv 0 \mod \text{sched} \).

Lemma 5. Let \( L \) be a partial schedule of cardinality at most \( \ell \). Then,

\[
y_L \in \text{span} \left( \left\{ y_S : \mid S \mid = \ell \text{ and } S \text{ is a partial schedule} \right\} \right).
\]
Proof. Assume that $|S| < \ell$ since otherwise we are done. Let $H \subseteq [m]$ such that $\delta_S(h) = 0$ for every $h \in H$, that is, $H$ is subset of machines that is not incident to the edges $S$ in the bipartite graph $G_S$, and $|H| = \ell - |S|$. Observe that since $S$ is a partial schedule, it is incident to exactly $|S|$ machines. Since $\sum_{C \in \mathcal{C}} y_{hC} \equiv 1 \mod \text{sched}$ for every $h \in H$, we have

$$y_S \equiv y_S \prod_{h \in H} y_{hC} \equiv \sum_{L \in \mathcal{C}^H} y_{S \cup L} \mod \text{sched},$$

where $\mathcal{C}^H$ is the set of partial schedules with domain $H$. In particular, for every $L \in \mathcal{C}^H$ we have that $S \cup L$ is a partial schedule, and $\deg(y_{S \cup L}) = |S| + \ell - |S| = \ell$. □

Let $Q^{\ell}_{\text{sched}}$ be the quotient ring of polynomials in $\mathbb{R}[y]$ with degree equal to $\ell$ that vanish in the ideal $\text{sched}$. Lemmas 4 and 5 above imply directly the following theorem.

**Theorem 5.** The quotient ring $Q^{\ell}_{\text{sched}}$ is spanned by $\{y_S : |S| = \ell \text{ and } S \text{ is a partial schedule}\}$.

### 3.4 Spanning sets of the invariant row subspace

In previous section we provided a reduced spanning set for the quotient ring vanishing in $\text{sched}$. In the following we construct spanning sets for the invariant row subspaces. Given a tableau $\tau_\lambda$ with shape $\lambda$, the hook($\tau_\lambda$) is the tableau with shape $(\lambda_1, 1, \ldots, 1) \in \mathbb{Z}^{m-\lambda_1+1}$, its first row is equal to the first row of $\tau_\lambda$ and the remaining elements of $\tau_\lambda$ fill the rest of the cells in increasing order over the rows. That part is called the tail of the hook, and we denote by tail($\tau_\lambda$) the elements of $[m]$ in the tail of hook($\tau_\lambda$), and row($\tau_\lambda$) = $[m] \setminus \text{tail}(\tau_\lambda)$, that is the elements in the first row of the tableau.

**Example 5.** Let $m = 7$ and consider the partition $\lambda = (4, 2, 1)$. The tableau $\tau_\lambda$ at the left has shape $\lambda$ and the tableau at the right is hook($\tau_\lambda$), with shape $(4, 1, 1, 1)$; row($\tau_\lambda$) = $\{1, 2, 7, 4\}$ and tail($\tau_\lambda$) = $\{3, 5, 6\}$.

\[
\begin{array}{cccc}
1 & 2 & 7 & 4 \\
5 & 6 & 3 \\
\end{array}
\quad
\begin{array}{cccc}
1 & 2 & 7 & 4 \\
3 & 5 & 6 \\
\end{array}
\]

The following lemma gives a spanning set for the row subspaces obtained from the hook tableau. We denote by $\text{sym}_{\text{hook}(\tau_\lambda)}$ the symmetrization respect to the row subgroup of hook($\tau_\lambda$),

$$\text{sym}_{\text{hook}(\tau_\lambda)}(f) = \frac{1}{|\text{R}_{\text{hook}(\tau_\lambda)}|} \sum_{\sigma \in \text{R}_{\text{hook}(\tau_\lambda)}} \sigma f. \quad (7)$$

**Lemma 6.** Given a tableau $\tau_\lambda$, the row subspace $W_{\tau_\lambda}$ of the quotient ring $Q^{\ell}_{\text{sched}}$ is spanned by

$$\{\text{sym}_{\text{hook}(\tau_\lambda)}(y_S) : |S| = \ell \text{ and } S \text{ is a partial schedule}\}. \quad (8)$$

12
Proof. The row subspace $W_{\tau\lambda}$ is spanned by $\{\text{sym}_{\text{hook}(\tau\lambda)}(yS) : |S| \leq \ell\}$ [59, Lemma 2]. By Corollary 5 the monomial basis is spanned by the partial schedules of size equal to $\ell$, so the lemma follows by linearity of the symmetrization operator.

In the row subgroup $R_{\text{hook}(\tau\lambda)}$, the elements of $[m]$ that are in the tail remain fixed. The rest of the elements on the first row are permuted arbitrarily. In particular, $R_{\text{hook}(\tau\lambda)} \cong S_{\lambda}$. Therefore, any permutation $\sigma$ in $R_{\text{hook}(\tau\lambda)}$ acts over a monomial $yS$ by separating de bipartite graph $G_S$ into those vertices in tail$(\tau\lambda)$ that are fixed by $\sigma$ and the rest in row$(\tau\lambda)$ that can be permuted.

Configuration profiles and extensions. Observe that bipartite graphs corresponding to different partial schedules are isomorphic if and only if the degree of every configuration is the same in both graphs. We say that a partial schedule is in $\gamma$-profile, with $\gamma : C \rightarrow \mathbb{Z}_+$, if for every $C \in C$ we have $\delta_S(C) = \gamma(C)$. Observe that a partial schedule in $\gamma$-profile has size $\sum_{C \in C} \gamma(C)$, quantity that we denote by $\|\gamma\|$. We denote by supp$(\gamma)$ the support of the vector $\gamma$, namely, $\{C \in C : \gamma(C) > 0\}$.

Definition 1. Given a partial schedule $T$, we say that a partial schedule $A$ over $[m] \setminus \mathcal{M}(T)$ is a $(T, \gamma)$-extension if $A$ is in $\gamma$-profile. We denote by $\mathcal{F}(T, \gamma)$ the set of $(T, \gamma)$-extensions. In particular, every $(T, \gamma)$-extension has size $\|\gamma\|$.

Example 6. Suppose that $m = 4$, $C = \{C_1, C_2\}$ and $T = \{(2, C_1), (3, C_2)\}$. For the profile $\gamma = (1, 1)$, we have that $\mathcal{F}(T, \gamma) = \{\{(1, C_1), (2, C_2)\}, \{(2, C_1), (1, C_2)\}\}$. For the profile $\mu = (1, 0)$, we have that $\mathcal{F}(T, \mu) = \{\{(1, C_1)\}, \{(4, C_1)\}\}$.

Given a partial schedule $T$ and a $\gamma$-profile, let $B_{T, \gamma}$ be the polynomial defined by

$$B_{T, \gamma} = \sum_{A \in \mathcal{F}(T, \gamma)} y_A,$$

if $\gamma \neq 0$, and 1 otherwise. In words, the polynomial above corresponds to sum over all those partial schedules in $\gamma$-profile that are not incident to $\mathcal{M}(T)$. The following theorem is the main result of this section.

Theorem 6. Let $\lambda \in \Lambda_\ell$ and a tableau $\tau\lambda$ of shape $\lambda$. Then, the row subspace $W_{\tau\lambda}$ of $Q^\text{sched}_T$ is spanned by

$$P^\lambda = \bigcup_{\omega = 0}^{\ell} \left\{y_T B_{T, \gamma} : T \text{ is partial schedule with } \mathcal{M}(T) = \text{tail}(\tau\lambda) \text{ and } \|\gamma\| = \omega \right\}.$$

Proof of Theorem 6. By Lemma 6 it is enough to check that the set of polynomials in (8) is spanned by those in (10). Let $S$ be a partial schedule of size $\ell$. Let tail$(S, \tau\lambda)$ be the subset of $S$ that is incident to the tail of the tableau, that is, $\{(i, C) \in S : i \in \text{tail}(\tau\lambda)\}$, and let row$(S, \tau\lambda) = S \setminus \text{tail}(S, \tau\lambda)$ be the edges of the partial schedule $S$ incident to the first row of the tableau.

Claim 1. $\text{sym}^\text{hook}(\tau\lambda)(yS) = y_{\text{tail}(S, \tau\lambda)} \cdot \text{sym}^\text{hook}(\tau\lambda)(y_{\text{row}(S, \tau\lambda)})$.

Observe that tail$(S, \tau\lambda)$ is a partial schedule over tail$(\tau\lambda)$. Similarly as we did in Lemma 5, the partial schedule incident to the tail can be completed to be in the span of partial schedules with domain equal to tail$(\tau\lambda)$, that is,

$$y_{\text{tail}(S, \tau\lambda)} \equiv y_{\text{tail}(S, \tau\lambda)} \prod_{h \in \text{tail}(\tau\lambda) \setminus \text{tail}(S, \tau\lambda)} \sum_{C \in C} y_h C \equiv \sum_{L \in \mathcal{C}(\text{tail}(\tau\lambda) \cup L)} y_{\text{tail}(S, \tau\lambda) \cup L} \mod \text{sched}$$
where $\mathcal{C}^{\text{tail}(\tau_\lambda)\setminus\text{tail}(S, \tau_\lambda)}$ is the set of partial schedules with domain $\text{tail}(\tau_\lambda) \setminus \text{tail}(S, \tau_\lambda)$. Thus, every partial schedule in the summation above have domain $\text{tail}(\tau_\lambda) \cup \text{tail}(S, \tau_\lambda) \setminus \text{tail}(S, \tau_\lambda) = \text{tail}(\tau_\lambda)$. Therefore, it is enough to check that exists a constant $\kappa$ such that

$$\text{sym}_{\text{row}(\tau_\lambda)}(y_{\text{row}(S, \tau_\lambda)}) = \kappa \cdot \mathcal{B}_{\text{tail}(\tau_\lambda), \gamma}$$

for some profile $\gamma$ with $\|\gamma\| = \ell - |\text{tail}(S, \tau_\lambda)|$. Recall that $|\text{tail}(S, \tau_\lambda)| \leq \ell$ since $\lambda \in \Lambda_\ell$. Let $\gamma$ be the profile of the partial schedule $\text{row}(S, \tau_\lambda)$. The equality follows since $\sigma \in \mathcal{R}_{\text{hook}(\tau_\lambda)} \cong S_{\text{row}(\tau_\lambda)}$, and that $\{(\sigma(i), C) : (i, C) \in \text{row}(S, \tau_\lambda)\}$ is a $(\text{tail}(\tau_\lambda), \gamma)$-extension for every permutation in $\sigma \in \mathcal{R}_{\text{hook}(\tau_\lambda)}$. The constant $\kappa$ is equal to $|\mathcal{R}_{\text{hook}(\tau_\lambda)}|$. \hfill $\square$

**Proof of Claim 1.** Observe that for every permutation $\sigma \in \mathcal{R}_{\text{hook}(\tau_\lambda)}$, we have

$$\sigma y_S = \prod_{(i, C) \in S} y_{\sigma(i)C} = \prod_{(i, C) \in \text{tail}(S, \tau_\lambda)} y_{\sigma(i)C} \prod_{(i, C) \in \text{row}(S, \tau_\lambda)} y_{\sigma(i)C} = y_{\text{tail}(S, \tau_\lambda)} \sigma y_{\text{row}(S, \tau_\lambda)},$$

since the permutation fixes the edges in $\text{tail}(S, \tau_\lambda)$. Therefore, symmetrizing yields to

$$\text{sym}_{\text{hook}(\tau_\lambda)}(y_S) = \frac{1}{|\mathcal{R}_{\text{hook}(\tau_\lambda)}|} \sum_{\sigma \in \mathcal{R}_{\text{hook}(\tau_\lambda)}} \sigma y_S = y_{\text{tail}(S, \tau_\lambda)} \cdot \frac{1}{|\mathcal{R}_{\text{hook}(\tau_\lambda)}|} \sum_{\sigma \in \mathcal{R}_{\text{hook}(\tau_\lambda)}} y_{\text{row}(S, \tau_\lambda)} = y_{\text{tail}(S, \tau_\lambda)} \cdot \text{sym}_{\text{hook}(\tau_\lambda)}(y_{\text{row}(S, \tau_\lambda)}).$$

\hfill $\square$

### 3.5 High-degree SoS pseudoexpectation: Proof of Theorem 1

We now have the algebraic ingredients to study the scheduling ideal and we detail next the SA pseudoexpectations from Theorem 3, that are the base for our lower bound. Recall that for every odd $k \in \mathbb{N}$, the hard instance $I_k$ has $m = 3k$ machines and the linear operators we consider are supported over partial schedules incident to a set of six so called *matching configurations*, $\{C_1, \ldots, C_6\}$. Consider the $\overline{E} : \mathbb{R}[y]_I/\mathcal{I}_E \rightarrow \mathbb{R}$ such that for every partial schedule $S$ of cardinality at most $k/2$,

$$\overline{E}(y_S) = \frac{1}{(3k)|S|} \prod_{j=1}^{6} (k/2) \delta_S(C_j),$$

where $(a)_b$ is the lower factorial function, that is, $(a)_b = a(a-1) \cdots (a-b+1)$, and $(a)_0 = 1$. The linear operator $\overline{E}$ is zero elsewhere. We state formally the main result that implies Theorem 1.

**Theorem 7.** For every odd $k \in \mathbb{N}$, the linear operator $\overline{E}$ is a degree-$[k/6]$ SoS pseudoexpectation for the configuration linear program in instance $I_k$ and $T = 1023$.

**Proof of Theorem 1.** For every odd $k$ the instance $I_k$ described in Section 3.1 is infeasible for $T = 1023$. By Theorem 7, the operator $\overline{E}$ is a degree-$[k/6]$ SoS pseudoexpectation, which in turns imply by Lemma 1 that there is no degree-$[k/6]$ SoS certificate of infeasibility. For an instance with $n$ jobs, let $k$ be the greatest odd integer such that $n = 15k + \ell$, with $\ell < 30$. The theorem follows by considering the instance $I_k$ above with $\ell$ dummy jobs of processing time equal to zero. \hfill $\square$
Theorem 3 guarantees that for every $k$ odd, $\tilde{\mathbb{E}}$ is a degree-$\lfloor k/2 \rfloor$ SA pseudoexpectation, and therefore a degree-$\lfloor k/6 \rfloor$ SA pseudoexpectation as well. In particular, properties (1) and (4) are satisfied. Since the configuration linear program is constructed from equality constraints, it is enough to check property (2) for high enough degree, in this case $\ell = \lfloor k/6 \rfloor$. To check property (2) we require a notion of conditional pseudoexpectations.

**Conditional pseudoexpectations.** Given a partial schedule $T$, consider the operator $\tilde{\mathbb{E}}_T : \mathbb{R}[y]/I_E \to \mathbb{R}$ such that

$$\tilde{\mathbb{E}}_T(y_S) = \frac{1}{(3k - |T|)!} \prod_{j=1}^{6} (k/2 - \delta_T(C_j)) \delta_S(C_j)$$

(12)

for every partial schedule $S$ over the machines $[m] \setminus M(T)$ and zero otherwise. Observe that if $T = \emptyset$ it corresponds to the linear operator $\tilde{\mathbb{E}}$ in (11). The following lemmas about the conditional pseudoexpectation in (12) are key for proving that $\tilde{\mathbb{E}}$ is a high-degree SoS pseudoexpectation. We state the lemmas and show how to conclude Theorem 1 using them. In particular, in Lemma 9 we prove a strong pseudo-independence property satisfied by the conditional pseudoexpectations and the polynomials (9) in the spanning set. We then prove the lemmas.

**Lemma 7.** The linear operator $\tilde{\mathbb{E}}$ is $S_m$-symmetric.

**Lemma 8.** Let $T$ be a partial schedule. Then, the following holds:

(a) If $S$ is a partial schedule and $T \cap S = \emptyset$, then $\tilde{\mathbb{E}}(y_T y_S) = \tilde{\mathbb{E}}_T(y_S) \tilde{\mathbb{E}}(y_T)$.

(b) If $S, R$ are two partial schedules such that $R \cap S = \emptyset$ and $T \cap (R \cup S) = \emptyset$, then

$$\tilde{\mathbb{E}}_T(y_R y_S) = \tilde{\mathbb{E}}_T(y_R) \tilde{\mathbb{E}}_{T \cup R}(y_S).$$

(c) Let $\nu$ be a profile with $\text{supp}(\nu) \subseteq \{C_1, \ldots, C_6\}$ and $|T| + ||\nu|| \leq k/2$. Then,

$$\tilde{\mathbb{E}}_T(B_{T, \nu}) = \prod_{j=1}^{6} \frac{1}{\nu(C_j)!} (k/2 - \delta_T(C_j))^{\nu(C_j)}.$$

**Lemma 9.** Let $T$ be a partial schedule and $\gamma, \mu$ a pair of configuration profiles with $|T| + ||\gamma|| + ||\mu|| \leq k/2$ and supp($\gamma$), supp($\mu$) $\subseteq \{C_1, \ldots, C_6\}$. Then,

$$\tilde{\mathbb{E}}_T(B_{T, \gamma} B_{T, \mu}) = \tilde{\mathbb{E}}_T(B_{T, \gamma}) \tilde{\mathbb{E}}_T(B_{T, \mu}).$$

(13)

**Proof of Theorem 7.** Let $\ell = \lfloor k/6 \rfloor$. Given a partition $\lambda \in \lambda_\ell$, consider the tableau $\tau_\lambda$ such that tail($\tau_\lambda$) = $[3k - \lambda_1]$ and row($\tau_\lambda$) = $[3k] \setminus [3k - \lambda_1]$. The partial schedules with domain $[3k - \lambda_1]$ can be identified with $C^{[3k - \lambda_1]}$, the set of functions from $[3k - \lambda_1]$ to $C$. In particular the spanning set in (10) is described by

$$\mathcal{P}^\lambda_\omega = \bigcup_{\omega=0}^\ell \left\{ y_T |_{B_{T, \gamma}} : T \in C^{[3k - \lambda_1]} \text{ and } ||\gamma|| = \omega \right\}.$$
To apply Lemma 3 we need to study the matrix \( \bar{E}(Z^\lambda) \). Recall that for \( T, S \in \mathcal{C}^{[3k-\lambda_1]} \) and profiles \( \gamma, \nu \) with \( \|\gamma\|, \|\nu\| \leq \ell \), the corresponding entry of the matrix \( \bar{E}(Z^\lambda) \) is given by

\[
\bar{E}\left(\text{sym}\left(y_T y_S \beta_T \gamma \beta_S \nu\right)\right) = \bar{E}\left(\text{sym}\left(y_{T \cup S} \beta_T \gamma \beta_S \nu\right)\right).
\]

By Lemma 7 the operator \( \bar{E} \) is symmetric, and therefore,

\[
\bar{E}\left(\text{sym}\left(y_{T \cup S} \beta_T \gamma \beta_S \nu\right)\right) = \bar{E}\left(y_{T \cup S} \beta_T \gamma \beta_S \nu\right).
\]

Since both \( T, S \) are partial schedules such that \( M(T) = M(S) \), we have that \( T \cup S \) is a partial schedule if and only if \( T = S \). Thus, the matrix \( \bar{E}(Z^\lambda) \) is block diagonal, with a block for each partial schedule \( T \in \mathcal{C}^{[3k-\lambda_1]} \). For every \( \Theta \) indexed by the elements of the spanning set above, we have then

\[
\Theta^\top \bar{E}(Z^\lambda) \Theta = \sum_{T \in \mathcal{C}^{[3k-\lambda_1]}} \sum_{\gamma: \|\gamma\| \leq \ell, \mu: \|\mu\| \leq \ell} \bar{E}\left(y_T \beta_T \gamma \beta_T \mu\right) \Theta_T \gamma \Theta_T \mu.
\]

Since \( |T| + \|\gamma\| + \|\mu\| \leq 3\ell \leq k/2 \) for every partial schedule \( T \) and profiles \( \gamma, \mu \) as above, by applying Lemma 8 (a) and Lemma 9 we obtain that

\[
\sum_{T \in \mathcal{C}^{[3k-\lambda_1]}} \sum_{\gamma: \|\gamma\| \leq \ell, \mu: \|\mu\| \leq \ell} \bar{E}\left(y_T \beta_T \gamma \beta_T \mu\right) \Theta_T \gamma \Theta_T \mu = \sum_{T \in \mathcal{C}^{[3k-\lambda_1]}} \bar{E}(y_T) \sum_{\gamma: \|\gamma\| \leq \ell, \mu: \|\mu\| \leq \ell} \bar{E}(\beta_T \gamma \beta_T \mu) \Theta_T \gamma \Theta_T \mu
\]

\[
= \sum_{T \in \mathcal{C}^{[3k-\lambda_1]}} \bar{E}(y_T) \sum_{\gamma: \|\gamma\| \leq \ell, \mu: \|\mu\| \leq \ell} \bar{E}(\beta_T \gamma) \bar{E}(\beta_T \mu) \Theta_T \gamma \Theta_T \mu,
\]

and by rearranging terms we conclude that

\[
\Theta^\top \bar{E}(Z^\lambda) \Theta = \sum_{T \in \mathcal{C}^{[3k-\lambda_1]}} \bar{E}(y_T) \left( \sum_{\gamma: \|\gamma\| \leq \ell} \bar{E}(\beta_T \gamma) \Theta_T \gamma \right)^2 \geq 0.
\]

**Proof of Lemma 7.** Given \( \sigma \in S_m \) and a partial schedule \( S, \bar{E}(\sigma y_S) = \bar{E}(y_{\sigma(S)}) \), where \( \sigma(S) = \{(\sigma(i), C) : (i, C) \in S\} \). In particular, since \( |S| = |\sigma(S)| \) and profile of \( S \) is the same profile of \( \sigma(S) \), it holds \( \bar{E}(y_S) = \bar{E}(\sigma y_S) \). Therefore, \( \bar{E}(y_S) = \frac{1}{m} \sum_{\sigma \in S_m} \bar{E}(\sigma y_S) = \bar{E}(\text{sym}(y_S)) \).

**Proof of Lemma 8.** Property b) implies a) by taking \( T = \emptyset \). One can check from the definition of the lower factorial that \( (x)_{a+b} = (x)_a(x-a)_b \). Since the partial schedules \( R, S \) and \( T \) are disjoint, it holds for every \( C \in \mathcal{C} \) that \( \delta_{R \cup S}(C) = \delta_R(C) + \delta_S(C) \) and \( \delta_{T \cup R}(C) = \delta_T(C) + \delta_R(C) \). Therefore,

\[
(3k - |T|)_{R \cup S} \cdot \bar{E}_T(y_{R \cup S}) = \prod_{j=1}^{6} \left( k/2 - \delta_T(C_j) \right) \delta_R(C_j) + \delta_S(C_j)
\]

\[
= \prod_{j=1}^{6} \left( k/2 - \delta_T(C_j) \right) \delta_R(C_j) \prod_{j=1}^{6} \left( k/2 - \delta_T(C_j) - \delta_R(C_j) \right) \delta_S(C_j)
\]

\[
= (3k - |T|)_{R} \cdot \bar{E}_T(y_{R}) \cdot (3k - |T| - |R|)_{S} \cdot \bar{E}_{T \cup R}(y_S),
\]
and the lemma follows since $(3k - |T|)_{|R|,|S|} = (3k - |T|)_{|R|} \cdot (3k - |T| - |R|)_{|S|}$.

We now prove property (c), that is more involved. First of all, observe that for every $H \in \mathcal{F}(T, \gamma)$ the value of $\tilde{E}_T(y_H)$ depends only on $T$ and the configuration profile $\gamma$. More specifically,

$$\tilde{E}_T(y_H) = \frac{1}{(3k - |T|)!} \prod_{j=1}^{6} (k/2 - \delta_T(C_j))! \gamma(C_j),$$

since $|H| = \|\gamma\|$ and $\delta_H(C_j) = \gamma(C_j)$ for every $j \in \{1, \ldots, 6\}$. Then, $\tilde{E}_T(B_{T, \gamma})$ equals $|\mathcal{F}(T, \gamma)|$ times the quantity above. The number of machines that can support a partial schedule $H$ that extend $T$ is $3k - |T|$, and since $|H| = \|\gamma\|$ the number of possible machine domains is

$$\left( \begin{array}{c} 3k - |T| \\ \|\gamma\| \end{array} \right).$$

Given a set of machines with cardinality $\|\gamma\|$, the number of partial schedules with domain equal to this set of machines and that are in configuration profile $\gamma$ are

$$\|\gamma\|! \prod_{j=1}^{6} \frac{1}{\gamma(C_j)!}.$$ 

Then, overall, the value of $\tilde{E}_T(B_{T, \gamma})$ equals to

$$\left( \begin{array}{c} 3k - |T| \\ \|\gamma\| \end{array} \right) \|\gamma\|! \frac{1}{(3k - |T|)!} \prod_{j=1}^{6} \frac{1}{\gamma(C_j)!} (k/2 - \delta_T(C_j))! \gamma(C_j)$$

$$\cdot \prod_{j=1}^{6} \frac{1}{\gamma(C_j)!} (k/2 - \delta_T(C_j))! \gamma(C_j)$$

$$= \prod_{j=1}^{6} \frac{1}{\gamma(C_j)!} (k/2 - \delta_T(C_j))! \gamma(C_j),$$

in the last step we used that for every real $x$ and non-negative integer $b$, it holds $(x-b)!/(x)_b = x!$. \qed

To prove Lemma 9 we obtain first a weaker version, that together with a polynomial decomposition in the scheduling ideal yields to the pseudoindependence result.

**Lemma 10.** Let $T$ be a partial schedule.

(a) If $\nu$ and $\xi$ are configuration profiles such that $\text{supp}(\nu) \cap \text{supp}(\xi) = \emptyset$ and $|T| + \|\nu\| + \|\xi\| \leq k/2$, then $\tilde{E}_T(B_{T, \nu}B_{T, \xi}) = \tilde{E}_T(B_{T, \nu})\tilde{E}_T(B_{T, \xi})$.

(b) If $\nu$ and $\xi$ are configuration profiles such that there exists $C \subseteq \{C_1, \ldots, C_6\}$ with $\text{supp}(\nu), \text{supp}(\xi) \subseteq \{C\}$, and $|T| + \|\nu\| + \|\xi\| \leq k/2$, then $\tilde{E}_T(B_{T, \nu}B_{T, \xi}) = \tilde{E}_T(B_{T, \nu})\tilde{E}_T(B_{T, \xi})$.

**Proof.** In both case if one of the profiles is zero then the conclusion follows. Then, in what follows assume that $\nu$ and $\xi$ are different from zero, and their support is contained in $\{C_1, \ldots, C_6\}$. Consider $\nu$ and $\xi$ satisfying the conditions in (a) and fix $A \in \mathcal{F}(T, \nu)$. Then,

$$\tilde{E}_T(y_{AB}B_{T, \xi}) = \sum_{B \in \mathcal{F}(T, \xi)} \tilde{E}(y_{AB}B) = \sum_{B \in \mathcal{F}(T \cup A, \xi)} \tilde{E}(y_{AB}B) + \sum_{B \in \mathcal{F}(T, \xi) \backslash \mathcal{F}(T \cup A, \xi)} \tilde{E}(y_{AB}B),$$

17
where the last equality holds since \( F(T \cup A, \xi) \subseteq F(T, \xi) \). For every term \( B \in F(T, \xi) \setminus F(T \cup A, \xi) \) we have that it is incident to at least one of the machines in \( G_A \). Since every machine in \( G_A \) is connected to a machine in \( \text{supp}(\nu) \subseteq C \setminus \text{supp}(\xi) \), it follows that \( A \cup B \) is not a partial schedule since at least one machine is connected to different configurations, and in consequence its pseudoexpectation is zero. Therefore, the second summation in the equality above is zero. Together with property (b) in Lemma 8 it implies that

\[
\tilde{E}_T(y_A B_{T, \xi}) = \sum_{B \in F(T \cup A, \xi)} \tilde{E}(y_A y_B) = \tilde{E}_T(y_A) \cdot \tilde{E}_{T \cup A}(B_{T \cup A, \xi}).
\]

Since \( \text{supp}(\nu) \cap \text{supp}(\xi) = \emptyset \), we have that for every \( C_j \in \text{supp}(x_i) \), \( \delta_{T \cup A}(C_j) = \delta_T(C_j) \). On the other hand, if \( C_j \notin \text{supp}(\xi) \) then \( (x)_{\xi(C_j)} = (x)_0 = 1 \) for every real \( x \). Overall, and together with Lemma 8, it holds that

\[
\tilde{E}_{T \cup A}(B_{T \cup A, \xi}) = \prod_{j=1}^{6} \frac{1}{\xi(C_j)!} (k/2 - \delta_{T \cup A}(C_j))_{\xi(C_j)} = \prod_{j \in \text{supp}(\xi)} \frac{1}{\xi(C_j)!} (k/2 - \delta_T(C_j))_{\xi(C_j)} = \tilde{E}_T(B_{T, \xi}).
\]

Together with the linearity of \( \tilde{E}_T \) we conclude (a).

Consider now \( \nu, \xi \) satisfying the conditions in (b), and let \( C \in \{ C_1, \ldots, C_6 \} \) the configuration that supports both profiles. Without loss of generality suppose that \( \nu(C) \geq \xi(C) \). For \( A \in F(T, \nu) \) and \( B \in F(T, \xi) \), we have that \( A \cup B \) is always a perfect matching since the profiles are supported in the same configuration. If \( B \subseteq A \), then the union has profile \( \nu \). Then, by Lemma 8 (c) we have

\[
\tilde{E}_T(y_A B_{T, \xi}) = \sum_{B \in F(T, \xi)} \tilde{E}_T(y_A y_B) = \sum_{B \in F(T, \xi) : B \subseteq A} \tilde{E}_T(y_A) + \sum_{B \in F(T, \xi) : B \setminus A \neq \emptyset} \tilde{E}_T(y_A y_B \setminus A)
\]

\[
= \tilde{E}_T(y_A) \left( \frac{\nu(C)}{\xi(C)} \right) + \sum_{B \in F(T, \xi) : B \setminus A \neq \emptyset} \tilde{E}_{T \cup A}(y_B \setminus A)
\]

If \( B \setminus A \neq \emptyset \), the union profile can be parameterized in \( |B \setminus A| = \omega \), and let \( \alpha_{\omega} \) be the profile such that \( \alpha_{\omega}(C) = \omega \) and zero otherwise. Thus,

\[
\sum_{B \in F(T, \xi) : B \setminus A \neq \emptyset} \tilde{E}_{T \cup A}(y_B \setminus A) = \sum_{\omega=1}^{\xi(C)} \left( \frac{\nu(C)}{\xi(C) - \omega} \right) \frac{(k/2 - \delta_T(C) - \nu(C))_{\omega}}{(3k - |T| - \nu(C))_{\omega}}
\]

\[
= \sum_{\omega=1}^{\xi(C)} \frac{1}{\omega!} \left( \frac{\nu(C)}{\xi(C) - \omega} \right) (k/2 - \delta_T(C) - \nu(C))_{\omega},
\]

and since \( (k/2 - \delta_T(C) - \nu(C))_0 = 1 \), and running the summation over \( A \in F(T, \nu) \) we obtain over all that

\[
\tilde{E}_T(B_{T, \nu} B_{T, \xi}) = \tilde{E}_T(B_{T, \nu}) \cdot \sum_{\omega=0}^{\xi(C)} \frac{1}{\omega!} \left( \frac{\nu(C)}{\xi(C) - \omega} \right) (k/2 - \delta_T(C) - \nu(C))_{\omega}.
\]
Claim 2. Let $a$ and $b$ be two non-negative integers such that $a \leq b$. Then, for every real $x$,
\[
\sum_{\omega=0}^{a} \frac{1}{\omega!} \binom{b}{a-\omega} (x-b)\omega = \frac{1}{a!}(x)_a.
\]
The claim applied in (14) for $x = k/2 - \delta_T(C)$, $a = \xi(C)$ and $b = \nu(C)$ yields the result, since
\[
\tilde{E}_T(B_{T,\nu}B_{T,\xi}) = \tilde{E}_T(B_{T,\nu}) \cdot \frac{1}{\xi(C)!}(k/2 - \delta_T(C))_{\xi(C)} = \tilde{E}_T(B_{T,\nu})\tilde{E}_T(B_{T,\xi}).
\]
The claim follows by the Chu-Vandermonde identity [3, p. 59-60],
\[
(x)_a = \sum_{\omega=0}^{a} \binom{a}{\omega} (x-b)_\omega (b)_{a-\omega} = a! \sum_{\omega=0}^{a} (x-b)_\omega \frac{(b)_{a-\omega}}{(a-\omega)!} = a! \sum_{\omega=0}^{a} (x-b)_{\omega} \frac{b}{a-\omega}.
\]

Proof of Lemma 9. Given a profile configuration $\gamma$ and $C_j \in \{C_1, \ldots, C_6\}$, we denote by $\gamma_j$ the profile that is zero for every $C \neq C_j$ and $\gamma_j(C_j) = \gamma(C_j)$. In the following, we prove that the following factorization holds:
\[
\tilde{E}_T(B_{T,\gamma}B_{T,\mu}) = \tilde{E}_T \left( \prod_{j=1}^{6} B_{T,\gamma_j}B_{T,\mu_j} \right),
\]
recalling that $B_{T,\xi} = 1$ if $\xi = 0$. Before checking that the decomposition above, we see how to conclude the lemma from that. Observe that by construction $\text{supp}(\gamma_j) \cap \text{supp}(\gamma_\ell) = \emptyset$ if $j \neq \ell$, and therefore by Lemma 10 (a), we have
\[
\tilde{E}_T \left( \prod_{j=1}^{6} B_{T,\gamma_j}B_{T,\mu_j} \right) = \prod_{j=1}^{6} \tilde{E}_T \left( B_{T,\gamma_j}B_{T,\mu_j} \right).
\]
Furthermore, since for every $j$ we have $\text{supp}(\gamma_j)$, $\text{supp}(\mu_j) \subseteq \{C_j\}$, by Lemma 10 (b) we have
\[
\prod_{j=1}^{6} \tilde{E}_T \left( B_{T,\gamma_j}B_{T,\mu_j} \right) = \prod_{j=1}^{6} \tilde{E}_T \left( B_{T,\gamma_j} \right) \tilde{E}_T \left( B_{T,\mu_j} \right).
\]
By using Lemma 10 (a) we can now reorder and group the elements in the right hand side,
\[
\prod_{j=1}^{6} \tilde{E}_T \left( B_{T,\gamma_j} \right) \tilde{E}_T \left( B_{T,\mu_j} \right) = \prod_{j=1}^{6} \tilde{E}_T \left( B_{T,\gamma_j} \right) \cdot \prod_{j=1}^{6} \tilde{E}_T \left( B_{T,\mu_j} \right)
\]
\[
= \tilde{E}_T \left( \prod_{j=1}^{6} B_{T,\gamma_j} \right) \cdot \tilde{E}_T \left( \prod_{j=1}^{6} B_{T,\mu_j} \right) = \tilde{E}_T \left( B_{T,\gamma} \right) \tilde{E}_T \left( B_{T,\mu} \right),
\]
where in the last equality we used the decomposition in (15) separately for $\gamma$ and $\mu$. We check now that the factorization in (15) is always valid. Let $S$ be a partial schedule disjoint from $T$ and with profile $\mu$ and let $C_j \in \text{supp}(\gamma)$. It is enough to check that
\[
\tilde{E}_T \left( B_{T,\gamma} y_S \right) = \tilde{E}_T \left( B_{T,\gamma_j} B_{T,\gamma-\gamma_j} y_S \right),
\]
since the factorization follows by the linearity of $\tilde{E}_T$ and by applying iteratively for every $C_j \in \{C_1, \ldots, C_6\}$ the above factorization. We have that

$$
\tilde{E}_T(B_T, y_S) = \tilde{E}_T \left( \sum_{A \in F(T, \gamma)} y_A y_S \right) = \tilde{E}_T \left( \sum_{B \in F(T, \gamma_j)} y_B \sum_{D \in F(T \cup B, \gamma - \gamma_j)} y_D y_S \right).
$$

Fix $B \in F(T, \gamma_j)$ and consider a set $D \in F(T, \gamma - \gamma_j) \setminus F(T \cup B, \gamma - \gamma_j)$. In particular, $D$ is in profile $\gamma - \gamma_j$ but is incident to at least one machine, say $\ell$, that is also incident to $B$. Since $B$ is in profile $\gamma_j$ and it has disjoint support from $\gamma - \gamma_j$, the above implies that machine $\ell$ is incident to different configurations, and therefore its pseudoexpectation value is equal to zero. That is the contribution to the pseudoexpectation value of the terms in $F(T, \gamma - \gamma_j) \setminus F(T \cup B, \gamma - \gamma_j)$ is is zero. Furthermore, since $F(T, \gamma - \gamma_j) \supseteq F(T \cup B, \gamma - \gamma_j)$, we have that for every $B \in F(T, \gamma_j)$,

$$
\tilde{E}_T \left( y_B \sum_{D \in F(T \cup B, \gamma - \gamma_j)} y_D y_S \right) = \tilde{E}_T \left( y_B \left( \sum_{D \in F(T \cup B, \gamma - \gamma_j)} y_D + \sum_{D \in F(T, \gamma - \gamma_j) \setminus F(T \cup B, \gamma - \gamma_j)} y_D \right) y_S \right)
$$

$$
= \tilde{E}_T \left( y_B \sum_{D \in F(T, \gamma - \gamma_j)} y_D y_S \right) = \tilde{E}_T(y_B B_{T, \gamma - \gamma_j} y_S).
$$

We conclude by summing over $B \in F(T, \gamma_j), S \in F(T, \mu)$ and using the linearity of $\tilde{E}_T$. 

\[\square\]

### 4  Upper bound: Breaking symmetries to approximate with SoS

In the previous section we showed that the action of the symmetric group is hard to tackle for the SoS method. In the following we show that we can obtain almost optimal relaxations in terms of integrality gap if we apply the SoS method after breaking symmetries in the ground formulation. Furthermore, the ground formulation we use is not the strong configuration linear program, but rather the weaker assignment linear program.

#### 4.1 Assignment Linear Program

A straightforward way to model the problem of makespan scheduling on identical machines is by an integer program called assignment linear program, which has a variables $x_{ij}$ modeling whether job $j$ is assigned to machine $i$. In this model, there are variables $x_{ij}$ indicating whether job $j$ is assigned to machine $i$. For a given guess on the optimal makespan $T$, consider the formulation assign($T$), given by

\[
\begin{align*}
\sum_{i \in M} x_{ij} & = 1 \quad \text{for all } j \in J, \\
\sum_{j \in J} x_{ij}p_j & \leq T \quad \text{for all } i \in M, \\
x_{ij} & \in \{0, 1\} \quad \text{for all } i \in M, \text{ for all } j \in J.
\end{align*}
\]
The assignment linear program corresponds to the linear relaxation where the last constraint is changed to \( x_{ij} \geq 0 \). With the additional constraint that \( T \geq \max_{j \in J} p_j \), the assignment linear program has an integrality gap of 2 [68].

The assignment linear program is invariant. In what follows we consider \( M = [m] \). The symmetric group \( S_m \) acts over the monomials in \( \mathbb{R}[x] \) according to \( \sigma x_{ij} = x_{\sigma(i)j} \), for every \( \sigma \in S_m \). The action extends linearly to \( \mathbb{R}[x]/I_E \), and assign\((T)\) is invariant under this action, that is, for every \( x \in \text{assign}(T) \) and every \( \sigma \in S_m \) we have \( \sigma x \in \text{assign}(T) \).

4.2 Symmetry breaking inequalities

In what follows, we introduce a set of constraints that guarantees every integer solution in assign\((T)\) to obey a specific order on the configurations over the machines. If we knew that a machine \( \ell \) is scheduled according to a given configuration \( C \), it should be that every machine smaller, and larger, than \( \ell \) is scheduled respecting the order on the configurations set. This is a way of breaking the machine symmetries since we are restricting the set of possible assignments.

Job-sizes partitions. We need some notation before introducing the symmetry breaking constraints. Suppose we have a partitioning \( J \) of the jobs set \( J \) into \( s \) parts, \( J = \{J_1, \ldots, J_s\} \). In principle, the partition is arbitrary. An example of such a partition is given in the following way. Suppose the job sizes are ordered from largest to smallest, that is \( p^1 > p^2 > \cdots > p^s \) where \( s = |\{p_j : j \in J\}| \) is the number of different job sizes. This particular case, where \( J_q = \{j \in J : p_j = p^q\} \) for every \( q \in [s] \), we call it the job-sizes partition. Given a partitioning of the jobs, a configuration \( C \) is a multiset of elements in \( \{1, \ldots, s\} \). Recall that for every \( q \in \{1, \ldots, s\} \), the multiplicity of \( q \) in \( C \), \( m(q, C) \), is the number of times that \( q \) appears repeated in \( C \).

**Example 7.** For a partition with \( s = 3 \) and \( C = \{1, 1, 1, 2\} \), we have \( m(1, C) = 3, m(2, C) = 1 \) and \( m(3, C) = 0 \).

As we did in the previous sections, we denote by \( C \) the set of all configurations. Observe that it coincides with the configuration notion introduced in Section 3.1 if we consider the job-sizes partition. We say that a configuration \( C \) is lexicographically larger than \( S \), and we denote \( C >_{\text{lex}} S \), if there exists \( q \in [s] \) such that \( m(\ell, C) = m(\ell, S) \) for all \( \ell < q \) and \( m(q, C) > m(q, S) \). The relation \( >_{\text{lex}} \) defines a total order over \( C \).

**Example 8.** The configuration \( C = \{1, 1, 1, 2, 2\} \) is lexicographically larger than \( S = \{1, 1, 1, 2, 3\} \) since \( m(1, C) = m(1, S) = 3 \) and \( m(2, C) = 2 > m(2, S) = 1 \).

![Figure 4](image-url)

Figure 4: Suppose the jobs in red are \( J_1 \) in the partitioning. The schedule at the right does not respect the lexicographic order since the job in yellow is in a part \( J_s \), with \( s > 1 \).
Symmetry breaking inequalities. Given a positive integer $B$ and a partitioning $J$ of the jobs, consider the program $\text{assign}(B, T)$ given by

$$
\text{assign}(T) \cap \bigcap_{i=1}^{m-1} \left\{ x \in \mathbb{R}^{M \times J} : \sum_{q=1}^{s} B^{s-q} \sum_{j \in J_q} (x_{ij} - x_{i(j+1)j}) \geq 0 \right\}.
$$

To avoid confusion we sometimes use the notation $\text{assign}(J, B, T)$ to emphasize that we are considering the program for the jobs set $J$. We also remark that the symmetry breaking constraints depend on the partitioning of $J$. Given a subset of jobs $K \subseteq J$ such that $\sum_{j \in K} p_j \leq T$, we denote by $\text{conf}(K)$ the configuration such that for every $q \in \{1, \ldots, s\}$, $m(q, \text{conf}(K)) = |K \cap J_q|$. We then say that $\text{conf}(K)$ is the configuration induced by $K$.

Example 9. Suppose we consider the job-sizes partitioning, and we have two different job-sizes, that is $s = 2$. We have three machines, $m = 3$. If we take $B = 10$, the symmetry breaking constraints are given by,

$$
10 \sum_{j \in J_1} (x_{1j} - x_{2j}) + \sum_{j \in J_2} (x_{1j} - x_{2j}) \geq 0,
$$

$$
10 \sum_{j \in J_1} (x_{2j} - x_{3j}) + \sum_{j \in J_2} (x_{2j} - x_{3j}) \geq 0.
$$

In the following we show that for sufficiently large, but polynomially sized $B$, every integer solution in the program $\text{assign}(B, T)$ obeys the lexicographic order on configurations over the machines. More specifically, given a feasible integer solution $x \in \text{assign}(T)$ and a machine $i \in M$, let $\text{conf}_i(x) \in C$ be the configuration defined by the number of jobs for each possible part that are scheduled in $i$ according to $x$, that is, for every $q \in \{1, \ldots, s\}$, $m(q, \text{conf}_i(x)) = \sum_{j \in J_q} x_{ij}$.

Theorem 8. There exists $B^* = O(|J|^2)$ such that for every integer solution $x \in \text{assign}(B^*, T)$ and for every machine $i \in M \setminus \{m\}$, we have $\text{conf}_i(x) \succeq_{\text{lex}} \text{conf}_{i+1}(x)$.

In general, $\text{assign}(B, T)$ is not $S_m$-invariant, but it is a valid formulation \footnote{The program $\text{assign}(B, T)$ does not contain every possible schedule of makespan at most $T$ since by breaking symmetries many feasible solutions are removed. The crucial part is that we retain a representative solution for each orbit.} for the problem of finding a schedule with makespan at most $T$. More specifically, we show that if there is a schedule with makespan at most $T$, then there exists an integral solution in $\text{assign}(B, T)$.

Lemma 11. Suppose there exists an integral feasible solution in $\text{assign}(T)$. Then, there exists $B = O(|J|^2)$ for which there exists an integral feasible solution in $\text{assign}(B, T)$.

We prove the statements above by introducing an intermediate result connecting the lexicographic order over the configurations and the symmetry breaking constraints. Given $B \in \mathbb{N}$, let $\mathcal{L}_B : C \rightarrow \mathbb{R}$ be the function such that for every configuration $C \in \mathcal{C}$,

$$
\mathcal{L}_B(C) = \sum_{q=1}^{s} B^{s-q} m(q, C).
$$

Recall that the lexicographic order over the configurations induces a total order over $\mathcal{C}$. We show that exists a polynomially sized value of $B$ for which $\mathcal{L}_B$ is a strictly increasing function, that is, if $C <_{\text{lex}} S$ then $\mathcal{L}_B(C) < \mathcal{L}_B(S)$. Using this result we then prove Theorem 8.
Lemma 12. Let $B^* = 1 + 2s \max_{q \in [s]} |J_q|$. For every $B > B^*$, we have that $\mathcal{L}_B$ is strictly increasing.

Proof of Theorem 8. Fix a machine $i \in M \setminus \{m\}$. Since $x$ is an integral solution in $\text{assign}(B, T)$, we have $\text{conf}_i(x), \text{conf}_{i+1}(x) \in \mathcal{C}$. The symmetry breaking constraints implies that

$$0 \leq \sum_{q=1}^{s} B^{s-q} (m(q, \text{conf}_i(x)) - m(q, \text{conf}_{i+1}(x))) = \mathcal{L}_B(\text{conf}_i(x)) - \mathcal{L}_B(\text{conf}_{i+1}(x)).$$

Applying Lemma 12 for $B = 1 + 2s \max_{q \in [s]} |J_q| = O(|J|^2)$ it holds that $\mathcal{L}_B$ is strictly increasing and therefore $\text{conf}_i(x) \geq_{\text{lex}} \text{conf}_{i+1}(x)$. \hfill \Box

Proof of Lemma 11. Since there exists a schedule of makespan at most $T$, there exists an integral solution $x \in \text{assign}(T)$. Since the lexicographic relation defines a total order over $\mathcal{C}$, there exists a permutation $\sigma \in S_m$ such that for every $i \in M \setminus \{m\}$, $\text{conf}_{\sigma(i)}(x) \geq_{\text{lex}} \text{conf}_{\sigma(i+1)}(x)$. Consider the integral solution $\tilde{x}$ obtained by permuting the solution according to $\sigma$, that is, $\tilde{x} = \sigma x$. Then, for every $i \in M \setminus \{m\}$ it follows that

$$\sum_{q=1}^{s} B^{s-q} \sum_{j \in J_q} (\tilde{x}_{ij} - \tilde{x}_{(i+1)j}) = \sum_{q=1}^{s} B^{s-q} (m(q, \text{conf}_{\sigma(i)}(x)) - m(q, \text{conf}_{\sigma(i+1)}(x)))
= \mathcal{L}_B(\text{conf}_{\sigma(i)}(x)) - \mathcal{L}_B(\text{conf}_{\sigma(i+1)}(x)) \geq 0,$$

where the last inequality holds by Lemma 12. We conclude that $\tilde{x} \in \text{assign}(B, T)$. \hfill \Box

We recall that having $B$ of polynomial size is relevant at the moment of solving the linear program $\text{assign}(B, T)$. In particular, the input size is $O(|J|^2 \cdot \log(B^*)) = O(\text{poly}(|J|))$, and so in can be solved in time $O(\text{poly}(|J|))$. In what follows, we use a superscript over the configurations to indicate its lexicographic order, that is, $C^1 > C^2 > C^3 > \cdots > C^{|\mathcal{C}|}$.

Proof of Lemma 12. Consider two configurations $C, S \in \mathcal{C}$ such that $C >_{\text{lex}} S$. Let $\tilde{q}$ be the smallest integer such that the multiplicities of the configurations are different, that is, $m(\ell, C) = m(\ell, S)$ for every $\ell < \tilde{q}$. For the sake of contradiction suppose that $m(\tilde{q}, C) < m(\tilde{q}, S)$. In particular, every term up to $\max\{0, \tilde{q}-1\}$ in the summation defining $\mathcal{L}_B(C) - \mathcal{L}_B(S)$ is equal to zero. By upper bounding the summation from $\min\{s, \tilde{q} + 1\}$ we obtain that

$$\sum_{q=\min\{s, \tilde{q}+1\}}^{s} B^{s-q} (m(q, C) - m(q, S)) \leq \sum_{q=\min\{s, \tilde{q}+1\}}^{\tilde{q}} B^{s-q} (|m(q, C)| + |m(q, S)|)
\leq \sum_{q=\min\{s, \tilde{q}+1\}}^{\tilde{q}} B^{s-q} \cdot 2|J_q| < B^* \cdot B^{s-\tilde{q}-1} \leq B^{s-\tilde{q}},$$

and since $m(\tilde{q}, S) - m(\tilde{q}, C) \geq 1$ it follows that

$$\sum_{q=\tilde{q}}^{\tilde{q}} B^{s-q} (m(q, C) - m(q, S)) < B^{s-\tilde{q}} + B^{s-\tilde{q}} (m(\tilde{q}, C) - m(\tilde{q}, S))
< B^{s-\tilde{q}} (1 + m(\tilde{q}, C) - m(\tilde{q}, S)) < 0,$$

yielding to the contradiction. \hfill \Box
4.3 Balanced partitionings

Recall that to obtain a degree-$\ell$ SoS pseudoexpectation one can solve a semidefinite program in dimension $|E|^{O(\ell)}$, where $|E|$ is the number of variables of the ground program \[55, 9\]. Each of these semidefinite programs are tightenings of the linear relaxation of assign($B, T$) and correspond to the Lasserre/SoS hierarchy approach \[41, 42\]. In this section we study the integrality gap of these relaxations by rounding SoS pseudoexpectations. More specifically, we show that low-degree SoS pseudoexpectations of assign($B, T$), with the degree depending only on the number of configurations in $C$ and the size of the partitioning $J$, can be rounded to obtain integral solutions with almost optimal makespan. We control the refinement of the partitionings to obtain better approximations.

Balanced partitionings. We say a partitioning $J$ is $\alpha$-balanced, with $\alpha \geq 1$, if for every $K, H \subseteq J$ such that $\text{conf}(K) = \text{conf}(H)$,

$$\sum_{j \in K} p_j \leq \alpha \sum_{j \in H} p_j.$$ 

Observe that the job-sizes partitioning is 1-balanced. Other parameter that plays a key role is the maximum number of jobs that can be scheduled in the same machine with makespan at most $T$, that is,

$$\lambda = \max \left\{ |K| : \sum_{j \in K} p_j \leq T \right\}.$$ 

For example, if we knew that $p_j \geq T/3$ for every $j \in J$, then $\lambda \leq 3$. Recall that the set of configurations depends on the partitioning and let $\tau(C) = 2\lambda |C|$. The following is the main result of this section.

**Theorem 9.** Consider a value $T > 0$ and an $\alpha$-balanced partitioning of $J$. Suppose there exists a degree-$\tau(C)$ SoS pseudoexpectation for assign($B^*, T$). Then, we can find in polynomial time an integral solution $x^{\text{lex}} \in \text{assign}(B, \alpha T)$.

If we go back to the hard instances shown in Section 3.1, for $T = 1023$ there is a constant number of configurations for all the instances $\{I_k\}_{k \in \mathbb{N}}$ if we consider the job-sizes partition, since there are only 15 different job sizes, and with value $O(1)$ all of them. Therefore, we have that $\tau(C) = O(1)$. Recall that for $T = 1023$ there is no feasible schedule for the instance and the job-sizes partition is 1-balanced. Therefore, for every odd $k \in \mathbb{N}$, by Theorem 9 the degree of a SoS pseudoexpectation in assign($B^*, T$) is upper bounded by a constant, and therefore there is a low-degree SoS certificate of infeasibility. We leave the full proof of Theorem 9 to Section 4.5. In the next section we also show how to obtain a polynomial time approximation scheme using Theorem refthm:gap-Las.

$\varepsilon$-Partitionings. An idea that has been frequently exploited for designing approximation schemes in scheduling and packing is to split the instances into long and short jobs. Then, each sub-instance is solved exactly or approximatedly by using suitable techniques, and the subsolutions are merged afterwards in order to provide a solution for the original problem. Rounding the numeric values of the instances plays a key role in this approaches since it allows to reduce the underlying combinatorics. In what follows, given $\varepsilon > 0$, we say that a job $j \in J$ is long if $p_j \geq \varepsilon \cdot T$, and it is short otherwise. The subset of long jobs is denoted by $J_{\text{long}}$ and the short jobs are $J_{\text{short}} = J \setminus J_{\text{long}}$. We consider a partitioning obtained by grouping jobs with a similar processing time. More specifically,
for every $q \in \{1, \ldots, (1 - \varepsilon)/\varepsilon^2\}$,

$$J_q = \left\{ j \in J_{\text{long}} : \left( \frac{1}{\varepsilon} + q \right) \varepsilon^2 T > p_j \geq \left( \frac{1}{\varepsilon} + q - 1 \right) \varepsilon^2 T \right\},$$

and we call this the $\varepsilon$-partitioning of the long jobs. We show next that this partitioning is arbitrarily close to being $1$-balanced.

**Lemma 13.** For every $\varepsilon > 0$, the $\varepsilon$-partitioning is $(1 + \varepsilon)$-balanced.

**Proof.** Consider $K, H \subseteq J$ such that $\text{conf}(K) = \text{conf}(H) = C$ for the $\varepsilon$-partitioning $\mathcal{J}$. In particular, for every $q \in \{1, \ldots, |\mathcal{J}|\}$ we have that $|K \cap J_q| = m(q, C) = |H \cap J_q|$, and so there exists a bijection $\varphi_q : K \cap J_q \rightarrow H \cap J_q$. Furthermore, for any pair of jobs $j, \ell \in J_q$ it holds

$$\frac{p_j}{p_\ell} \leq \frac{(1/\varepsilon + q) \varepsilon^2 T}{(1/\varepsilon + q - 1) \varepsilon^2 T} \leq \frac{1/\varepsilon + 1}{1/\varepsilon} = 1 + \varepsilon,$$

since the function $(1/\varepsilon + q)/(1/\varepsilon + q - 1)$ is strictly decreasing in $[1, +\infty)$. Therefore, for every $q \in \{1, \ldots, |\mathcal{J}|\}$ it holds that $\sum_{j \in K \cap J_q} p_j \leq (1 + \varepsilon) \sum_{j \in H \cap J_q} p_{\varphi_q(j)} = (1 + \varepsilon) \sum_{j \in H \cap J_q} p_j$, and we conclude that

$$\sum_{j \in K} p_j = \sum_{q=1}^{J_q} \sum_{j \in K \cap J_q} p_j \leq (1 + \varepsilon) \sum_{q=1}^{J_q} \sum_{j \in H \cap J_q} p_{\varphi_q(j)} = (1 + \varepsilon) \sum_{j \in J_q} p_j.$$

### 4.4 SDP based approximation scheme: Proof of Theorem 2

Consider assign($B^*, T$) obtained from the $\varepsilon$-partitionings above and according to Theorem 8. Then, using a binary search procedure we look for the smallest $T$ such that there exists a degree-$\tau(C)$ SoS pseudoexpectation for the long jobs. Invoking Theorem 9 we then obtain a schedule for the long jobs. The short jobs are scheduled greedily.

**Algorithm 1**

**Input:** A scheduling instance and $T \geq 1/m \sum_{j \in J} p_j$.

**Output:** A schedule with makespan at most $(1 + \varepsilon)T$ if there exists a schedule for $J_{\text{long}}$ with makespan at most $T$; *infeasible* otherwise.

1. For all $(i, j) \in M \times J$, initialize $x_{ij} \leftarrow 0$.
2. Consider the $(\varepsilon/2)$-partitioning of $J_{\text{long}}$ and $B^* = 1 + 4\varepsilon^2 |J_{\text{long}}|$.
3. if there exists a degree-$\tau(C)$ SoS pseudoexpectation for assign($J_{\text{long}}, B^*, T$) then
   4. Construct the Lex-schedule $x_{\text{lex}}$ of $J_{\text{long}}$; for all $(i, j) \in M \times J_{\text{long}}$, $x_{ij} \leftarrow x_{ij}^{\text{lex}}$.
   5. while $J_{\text{short}} \neq \emptyset$ do
      6. Pick $k \in J_{\text{short}}$, and let $i \in M$ such that $i \in \arg\min_{\ell \in M} \sum_{j \in J} p_j \ell_{ij}$.
      7. update $x_{ik} \leftarrow 1$ and $J_{\text{short}} \leftarrow J_{\text{short}} \setminus \{k\}$.
   8. Return $x$.
9. else return infeasible.

**Proposition 1.** If there exists a degree-$\tau(C)$ SoS pseudoexpectation for assign($J_{\text{long}}, B^*, T$), the Algorithm 2 returns a schedule with makespan at most $(1 + \varepsilon)T$. 

25
Proof. By Theorem 9, since there exists a \( \tau(C) \)-degree SoS pseudoexpectation for assign \( (J_{\text{long}}, B^*, T) \) we have that \( \lambda_{\text{lex}} \) is an integral schedule for \( J_{\text{long}} \) with makespan at most \( T \). In this case, jobs in \( J_{\text{short}} \) are assigned according to the list scheduling algorithm, for which we include the analysis only for completeness. Let \( k \in J_{\text{short}} \) and let \( i \in M \) such that \( x_{ik} = 1 \). Since \( T > 1/m \sum_{j \in J \setminus \{k\}} p_j \), it follows that \( T > \sum_{j \in J} p_j x_{ij} \) since this value is minimized at \( i \in M \). Therefore, the load of machine \( i \) after scheduling job \( k \) is upper bounded by

\[
p_k + \sum_{j \in J : x_{ij} = 1} p_j < \frac{\varepsilon}{2} T + \left( 1 + \frac{\varepsilon}{2} \right) T = (1 + \varepsilon) T.
\]

\( \square \)

Approximation. Using Algorithm 1 the approximation scheme is constructed in a standard way. We perform a binary search procedure to find the smallest integer value of \( T \) such that there exists a degree-\( \tau(C) \) SoS pseudoexpectation for assign \( (J_{\text{long}}, B^*, T) \). In order to do that, it is enough to consider the lower bound \( 1/m \sum_{j \in J} p_j \) on the optimal makespan, and the upper bound \( [1/m \sum_{j \in J} p_j] + \max_{j \in J} p_j / \varepsilon \). For such value of \( T \), thanks to Theorem 1 we obtain a schedule with makespan at most \((1 + \varepsilon) T \). In particular, every optimal schedule induces a degree-\( \tau(C) \) SoS pseudoexpectation for assign \( (J_{\text{long}}, B^*, C_{\text{max}}) \), and therefore \( T \) is a lower bound for \( C_{\text{max}} \). We conclude that the makespan of this schedule is at most \((1 + \varepsilon) \cdot T \leq (1 + \varepsilon) \cdot 2C_{\text{max}} \).

Running time. The smallest job size in \( J_{\text{long}} \) is \((\varepsilon/2) T \), so we have \( \lambda \leq 2/\varepsilon \). The size of \( C \) can also be upper bounded by \((1 + 2/\varepsilon) 4/\varepsilon^2 \), since not more than \( 2/\varepsilon \) jobs can be allocated to single machine and the size of the partition is less than \( 4/\varepsilon^2 \). Since the running time of the algorithm is dominated by finding a degree-\( \tau(C) \) pseudoexpectation, the algorithm runs in polynomial time.

4.5 Pseudoexpectation rounding

Preliminaries. Before proceeding with the proof we need to introduce some properties that are satisfied by the SoS pseudoexpectations linear operators. Similarly to other convex hierarchies such as SA or LS\(_+\), for high-enough degree one can obtain an actual probability distribution over the integral solutions of the ground program [42, 66, 49]. We revisit a stronger result that implies this property as a corollary, known as the Decomposition Theorem [33]. This is a structural difference with other hierarchies, and makes the SoS hierarchy stronger than others. We make an exposition that is self-contained in the context of the assignment linear program. The following properties are very standard and we derive them using the pseudoexpectations approach.

Proposition 1. Let \( \bar{E} \) be a degree-\( \ell \) SoS pseudoexpectation of assign \( (B, T) \). Then, the following holds:

(a) For every \( I \subseteq M \times J \) with \( |I| \leq \ell/2 \), we have \( \bar{E}(x_I) \in [0, 1] \).

(b) For every \( L \subseteq M \times J \) with \( |L| \leq \ell/2 \) and \( I \subseteq L \), we have \( \bar{E}(x_L) \leq \bar{E}(x_I) \).

Proof. We check (a) first. By property (2) of the pseudoexpectations we have that \( 0 \leq \bar{E}((x_0 - x_1)^2) = \bar{E}(x_0) + \bar{E}(x_1) - 2 \cdot \bar{E}(x_1, 1) - \bar{E}(x_1) \), which implies that \( \bar{E}(x_I) \leq 1 \). Similarly, we have that \( 0 \leq \bar{E}(x_I^2) = \bar{E}(x_I) \), and therefore \( \bar{E}(x_I) \in [0, 1] \). To check (b), property (2) guarantees that

\[
0 \leq \bar{E}((x_I - x_L)^2) = \bar{E}(x_I) + \bar{E}(x_L) - 2 \cdot \bar{E}(x_I, x_L) - \bar{E}(x_I) - \bar{E}(x_L),
\]

where the last step follows since \( I \subseteq L \) and then \( x_I x_L = x_L \). We conclude that \( \bar{E}(x_L) \leq \bar{E}(x_I) \). \( \square \)
One of the key tools in our rounding algorithm is the notion of pseudoexpectation conditioning. Consider a degree-\(\ell\) pseudoexpectation \(\bar{E}\), let \(i \in M\) be a machine and \(K \subseteq J\) such that

\[
\bar{E}\left( \prod_{j \in K} x_{ij} \prod_{j \in J \setminus K} (1 - x_{ij}) \right) > 0. \tag{18}
\]

Observe that the polynomial above in (18) is equal to 1 if and only \(x_{ij} = 1\) for every \(j \in K\) and \(x_{ij} = 0\) for every \(j \in J \setminus K\). That is, machine \(i\) is scheduled integrally with the jobs in \(K\). For simplicity, we call \(\phi_{i,K} = \prod_{j \in K} x_{ij} \prod_{j \in J \setminus K} (1 - x_{ij})\). The \((i,K)\)-conditioning of \(\bar{E}\) corresponds to the linear operator over \(\mathbb{R}[x]/I_E\) defined by

\[
\bar{E}_{i,K}(x_I) = \frac{\bar{E}(x_I \phi_{i,K})}{\bar{E}(\phi_{i,K})}, \tag{19}
\]

for every \(I \subseteq M \times J\). Intuitively, the \((i,K)\)-conditioning is the pseudoexpectation value obtained conditioned on the event that machine \(i\) is scheduled integrally with the jobs in \(K\). The following property justifies the intuition, since it decomposes the pseudoexpectation as a convex combination of conditionings, which is the case for actual probability measures.

**Lemma 14.** Let \(\bar{E}\) be a degree-\(\ell\) pseudoexpectation with \(\ell \geq 2\lambda\) and consider a machine \(i \in M\). Then, the following holds:

(a) If \(\bar{E}(\phi_{i,K}) > 0\), then \(|K| \leq \lambda\).

(b) \(\bar{E}_{i,K}(x_{ij}) = 1\) for every \(j \in K\) and \(\bar{E}_{i,K}(x_{ij}) = 0\) for every \(j \in J \setminus K\).

(c) If there exists \(H \subseteq J\) such that \(\bar{E}(\phi_{i,H}) > 0\), then \(\sum_{K \subseteq J, \bar{E}(\phi_{i,K}) > 0} \bar{E}(\phi_{i,K}) \cdot \bar{E}_{i,K}\).

**Proof.** Applying the Mobius inversion we have that for every \(K \subseteq J\),

\[
\prod_{j \in K} x_{ij} \prod_{j \in J \setminus K} (1 - x_{ij}) = \sum_{H \subseteq J, K \subseteq H} (-1)^{|H \setminus K|} \prod_{j \in H} x_{ij}. \tag{20}
\]

If \(|K| > \lambda\) we have \(\sum_{j \in J} p_j > T\), and therefore \(\bar{E}(\prod_{j \in K} x_{ij}) = 0\). Since every term in the summation in (20) contains \(K\), by Proposition 1 we have \(\bar{E}(\prod_{j \in H} x_{ij}) = 0\) and by linearity of \(\bar{E}\) we conclude that \(\bar{E}(\phi_{i,K}) = 0\). That proves (a). For every \(j \in J\) we have that \(\bar{E}_{i,K}(x_{ij}) = \bar{E}(x_{ij} \phi_{i,K})/\bar{E}(\phi_{i,K})\). If \(j \in K\), \(x_{ij} \phi_{i,K} = \phi_{i,K}\), therefore \(\bar{E}_{i,K}(x_{ij}) = 1\). If \(j \in J \setminus K\), \(x_{ij} \phi_{i,K} = \prod_{t \in K} x_{it} \prod_{t \in J \setminus (K \cup \{j\})} (1 - x_{it}) \cdot x_{ij} (1 - x_{ij})\), and since \(\bar{E}(x_{ij}(1 - x_{ij})) = 0\) we conclude that \(\bar{E}_{i,K}(x_{ij}) = 0\). That proves (b). To prove (c), we verify that a stronger statement holds,

\[
\sum_{K \subseteq J} \phi_{i,K} = 1. \tag{21}
\]
In particular, by linearity follows that \( \sum_{K \subseteq J} \tilde{E}(\phi_{i,K}) = 1 \), and for every \( I \subseteq M \times J \),

\[
\tilde{E}(x_I) = \sum_{K \subseteq J} \tilde{E}(x_I \phi_{i,K}) = \sum_{K \subseteq J : E(\phi_{i,K}) > 0} \tilde{E}(\phi_{i,K}) \cdot \tilde{E}_{i,K}(x_I),
\]

which concludes (c). To check that (21) holds, observe that for every \( K, L \subseteq J \), \( \phi_{i,K} \phi_{i,L} = 0 \) if \( K \neq L \). Therefore, \( \sum_{K \subseteq J} \phi_{i,K}^2 = \sum_{K \subseteq J} \sum_{L \subseteq J} \phi_{i,K} \phi_{i,L} = \sum_{K \subseteq J} \phi_{i,K} \), and that implies \( \sum_{K \subseteq J} \phi_{i,K} \in \{0,1\} \). Since there exists \( H \subseteq J \) such that \( \tilde{E}(\phi_{i,K}) > 0 \), it is necessary that \( \sum_{K \subseteq J} \phi_{i,K} = 1 \).

We now state the Decomposition Theorem adapted to the assignment linear program in the language of pseudoexpectations. It was originally introduced using the moments approach, but they are equivalent and we refer to [63] for a proof and a detailed exposition of the SoS hierarchy.

**Theorem 10 ([33]).** Let \( \tilde{E} \) be a degree-\( \ell \) SoS pseudoexpectation of assign\((B, T)\), with \( \ell \geq 2\lambda \). Then, for every machine \( i \in M \) and a subset of jobs \( K \subseteq J \) such that \( \tilde{E}(\phi_{i,K}) > 0 \), the operator \( \tilde{E}_{i,K} \) is a degree-(\( \ell - 2\lambda \)) SoS pseudoexpectation of assign\((B, T)\).

We say that \( x \) is integral at machine \( i \in M \) if for every \( j \in J \) we have \( x_{ij} \in \{0,1\} \). This also extends to pseudoexpectations: we say that \( \tilde{E} \) is integral at machine \( i \) if \( \tilde{E}(x_{ij}) \in \{0,1\} \) for every \( j \in J \). Lemma 14 guarantees that a conditioning \( \tilde{E}_{i,K} \) is integral for machine \( i \) and this machine is scheduled with exactly the jobs in \( K \), when \( \tilde{E}(\phi_{i,K}) > 0 \). In our algorithm we iteratively decompose the current pseudoexpectation according to the above conditionings. Every time we perform this step we obtain a machine scheduled integrally, and therefore in order to progress we require that machine to remain integral along the execution.

**Proposition 2.** Let \( \tilde{E} \) be a degree-\( \ell \) pseudoexpectation of assign\((B, T)\), with \( \ell \geq 2\lambda \), and let \( h \in M \) be a machine such that \( \tilde{E}(x_{h,j}) \in \{0,1\} \) for every \( j \in J \). Let \( i \in M \) and \( K \subseteq J \) such that \( \tilde{E}_{i,K}(\phi_{i,K}) > 0 \). Then, \( \tilde{E}_{i,K}(x_{h,j}) = \tilde{E}(x_{h,j}) \in \{0,1\} \) for every \( j \in J \).

**Proof.** Let \( j \in J \) such that \( \tilde{E}(x_{h,j}) = 0 \). Observe that \( \tilde{E}_{i,K}(x_{h,j}) \leq \tilde{E}(x_{h,j} \phi_{i,K}) \), and by inclusion-exclusion and Proposition 1 it follows that \( \tilde{E}_{i,K}(x_{h,j}) = 0 \). On the other hand, if \( \tilde{E}(x_{h,j}) = 1 \), since \( \sum_{\ell \in M} \tilde{E}(x_{\ell,j}) = 1 \) it follows that \( \tilde{E}(x_{\ell,j}) = 0 \) for every \( \ell \neq h \). By using the argument above and since by Theorem 10 the restriction of \( \tilde{E}_{i,K} \) to the monomials of degree 1 is feasible in assign\((B, T)\), it follows that \( \tilde{E}_{i,K}(x_{\ell,j}) = 0 \) for every \( \ell \neq h \) and therefore \( \tilde{E}_{i,K}(x_{h,j}) = 1 - \sum_{\ell \in M} \tilde{E}_{i,K}(x_{\ell,j}) = 1 \).

**Overview of the rounding algorithm.** Consider a partitioning of the jobs that is \( \alpha \)-balanced. If we start from a high enough level of the hierarchy, we get at the end of the procedure a solution that is feasible for assign\((B^*, T)\), and therefore, the configurations of the integral machines have to obey the lexicographic order. The algorithm consist of two phases. In Phase 1, we use the solution obtained from high enough level of the hierarchy to find the last machine which is fractionally scheduled according to configuration \( C^1 \) using the Decomposition Theorem, and pick the corresponding conditioning pseudoexpectation. We then proceed by finding the last machine scheduled fractionally according to \( C^2 \) in the pseudoexpectation conditioning, and so on, for every configuration \( C^k \). We end up with a pseudoexpectation that is integral for all these machines, and it respects the lexicographic order. The number of conditioning steps is upper bounded by the number of configurations. In Phase 2, we greedily construct the schedule for the rest of the machines. The correctness
of Phase 2 is guaranteed by certifying the feasibility of a certain transportation problem. We call the schedule obtained in this way the lexicographic schedule, \( x^{\text{lex}} \).

**Algorithm 2**

**Input:** A degree-\( \tau(C) \) SoS pseudoexpectation \( \tilde{E} \) of assign\((B,T)\).

**Output:** A schedule \( x^{\text{lex}} \) with makespan at most \( \alpha T \).

1. For all \((i,j) \in M \times J\), initialize \( x_{ij}^{\text{lex}} \leftarrow 0\), \( \ell \leftarrow 0 \) and \( \tilde{E}^0 \leftarrow \tilde{E} \).

2: \( \triangleright \) **Phase 1:** Inducing integrality of machines.

3: \textbf{for} \( \ell = 1 \) to \(|C|\) \textbf{do}

4: \hspace{1em} Let \( M^\ell = \{i \in M : \text{exists } K \subseteq J \text{ with } \tilde{E}^\ell(\phi_{i,K}) > 0 \text{ and conf}(K) = C^\ell \} \).

5: \hspace{1em} \textbf{if} \( M^\ell \neq \emptyset \) \textbf{then}

6: \hspace{2em} let \( i^\ell = \max M^\ell \) and \( K^\ell \subseteq J \) such that \( \text{conf}(K^\ell) = C^\ell \) and \( \tilde{E}^\ell(\phi_{i^\ell,K^\ell}) > 0 \),

7: \hspace{2em} for every \( j \in K^\ell \), let \( x_{i^\ell j}^{\text{lex}} \leftarrow 1 \),

8: \hspace{2em} update the pseudoexpectation, \( \tilde{E}^{\ell+1} \leftarrow \tilde{E}^{\ell}_{i^\ell,K^\ell} \).

9: \hspace{1em} \textbf{else} \( i^\ell = \infty \)

10: \hspace{1em} Reset \( \ell \leftarrow 1 \), \( i^0 = 0 \).

11: \( \triangleright \) **Phase 2:** Extending the solution greedily.

12: \textbf{for} \( \ell = 1 \) to \(|C|\) \textbf{do}

13: \hspace{1em} \textbf{if} \( i^\ell < \infty \) \textbf{then}

14: \hspace{2em} \textbf{for} \( \max\{i^q : 0 \leq q < \ell, i^q < \infty\} < i < i^\ell \) \textbf{do}

15: \hspace{3em} let \( K^i \subseteq J \setminus \{j \in J : \text{exists } i \in M \text{ with } x_{ij}^{\text{lex}} = 1\} \) where \( \text{conf}(K^i) = C^\ell \);

16: \hspace{3em} for every \( j \in K^i \), let \( x_{ij}^{\text{lex}} \leftarrow 1 \).

17: Return \( x^{\text{lex}} \).

**Lemma 15.** Let \( \tilde{E} \) be a degree-\( \ell \) SoS pseudoexpectation of assign\((B,T)\), with \( \ell \geq 2\lambda \) and \( i,h \in M \) two machines with \( i < h \). Suppose that \( \tilde{E} \) is integral for machine \( h \). Then, for every \( K \) such that \( \tilde{E}(\phi_{i,K}) > 0 \), we have \( \text{conf}(K) \geq_{\text{lex}} \text{conf}_h(\tilde{E}) \).

**Proof.** Let \( K \subseteq J \) with \( \tilde{E}(\phi_{i,K}) > 0 \). By Theorem 10 we have that the restriction of \( \tilde{E}(\phi_{i,K}) \) to degree 1 monomials is in assign\((B,T)\), \( \tilde{E}_{i,K}(x_{ij}) \in \{0,1\} \) for all \( j \in J \) and \( \text{conf}(K) = \text{conf}_i(\tilde{E}_{i,K}) \). In particular, from the symmetry breaking constraints it follows that \( L_B(\text{conf}_i(\tilde{E}_{i,K})) \geq L_B(\text{conf}_h(\tilde{E})) \), since by Proposition 2 we have that \( \tilde{E}_{i,K} \) remains integral at \( h \) and in the same configuration than \( \tilde{E} \). The function \( L_B \) is strictly increasing, and therefore \( \text{conf}(K) = \text{conf}_i(\tilde{E}_{i,K}) \geq_{\text{lex}} \text{conf}_h(\tilde{E}) \). It is guaranteed by the algorithm that machine \( m \) is integral at the end of Phase 1. We prove next that at the end of every iteration of Phase 1, the machines between other two that have been induced to be integral by the conditionings, are all fractionally scheduled according to the same configuration. In particular, it guarantees that the greedy approach of Phase 2 works. \( \square \)
Leaves to be scheduled in Phase 2, the maximum lexicographic configuration.

Lemma 16. Let $\ell \in \{1, \ldots, |C|\}$. For every $i \in M$ such that $i_{\ell} = \max\{i^q : 0 \leq q < \ell, i^q < \infty\} < i \leq i_{\ell}$ and every $q \in \{1, \ldots, s\}$, $\sum_{j \in I_q} \tilde{E}^\ell(x_{ij}) = m(q, C^\ell)$.

Proof. Let $i \in M$ be a machine such that $i_{\ell} < i \leq i_{\ell}$. By Lemma 15, for every $K \subseteq J$ such that $E^\ell(\phi_i, K) > 0$ it holds that $conf(K) \succeq_{lex} C^\ell$, but we show next that they are all equalities. Suppose there exists $K \subseteq J$ such that $C^t = conf(K) >_{lex} C^\ell$ for some $t < \ell$. That would imply that $i \leq i_{\ell} < \infty$, contradicting that $i_{\ell} < i$. By Lemma 14, for every $q \in \{1, \ldots, s\}$ it holds that

$$\sum_{j \in I_q} \tilde{E}^\ell(x_{ij}) = \sum_{j \in I_q} \sum_{K \subseteq \bar{J}, E^\ell(\phi_i, K) > 0} E^\ell(\phi_i, K) \cdot \tilde{E}^\ell(x_{ij}) = \sum_{K \subseteq \bar{J}, E^\ell(\phi_i, K) > 0} E^\ell(\phi_i, K) \cdot \sum_{j \in I_q} \tilde{E}^\ell(x_{ij}) = \sum_{K \subseteq \bar{J}, E^\ell(\phi_i, K) > 0} E^\ell(\phi_i, K) \cdot m(q, C^\ell) = m(q, C^\ell).$$

Proof of Theorem 9. For every $\ell \in \{1, \ldots, |C|\}$, let $N^\ell = \{i \in M : i_{\ell} < i < i_{\ell}\}$. Observe that some of these sets could be equal to the $\emptyset$. Thanks to Lemma 16, at the end of Phase 1 we obtain a pseudoexpectation $\tilde{E}^{\ell}$, that evaluated in the degree 1 monomials satisfies every constraint of assign($B, T$) and it is integral for every machine in $M \setminus \cup_{\ell=1}^{|C|} N^\ell$. Every machine in $N^\ell$ is fractionally scheduled according to configuration $C^\ell$, for all $\ell \in \{1, \ldots, |C|\}$. We now see how we construct a schedule with makespan at most $\alpha \cdot T$ for the jobs that have not been scheduled in Phase 1, and using only the machines in $\tilde{M} = \cup_{\ell=1}^{|C|} N^\ell$.

Left $J$ be all the jobs that have not been scheduled in Phase 1. Consider the bipartite graph with nodes given by the jobs $\tilde{J}$ on one side, and the other side are the set of nodes $\mathcal{R} = \tilde{M} \times \{1, \ldots, s\}$. There is an edge between $j \in \tilde{J}$ and $(i, q) \in \mathcal{R}$ if $\tilde{E}^{\ell}(x_{ij}) > 0$ and $j \in J_q$. We consider a transportation problem where the offer of every node in $\tilde{J}$ is exactly 1, and the demand of a node $(i, q)$ is equal to $m(q, C^\ell)$ if $i \in N^\ell$. By construction the total offer equals to total demand and $\tilde{E}^{\ell}$ evaluated in the monomials of degree 1 is a fractional solution to this problem. Therefore, since this is a feasible transportation problem, the integrality of the flow formulation implies that exists an integral solution and a way of implementing Phase 2.

Given $\ell \in \{1, \ldots, |C|\}$ with $i_{\ell} < \infty$, in Phase 2 we have that every machine $i \in N^\ell$ is such that $conf(K^i) = C^\ell = conf(K)$, where $K^i$ are the jobs scheduled to $i$ in $x^{\text{lex}}$, and $K$ the jobs scheduled
to $i^\ell$. The partitioning of the jobs is $\alpha$-balanced, so we have that
\[
\sum_{j \in J} p_j x_{i_j}^{\text{lex}} = \sum_{j \in K} p_j \leq \alpha \sum_{j \in K} p_j = \alpha \sum_{j \in J} p_j x_{i_j}^{\text{lex}} \leq \alpha T,
\]
since the load at machine $i^\ell$ is at most $T$ thanks to Phase 1. That concludes the proof. \qed

References

[1] N. Alon, Y. Azar, G. J. Woeginger, and T. Yadid. Approximation schemes for scheduling. In SODA, pages 493–500, 1997.

[2] N. Alon, Y. Azar, G. J. Woeginger, and T. Yadid. Approximation schemes for scheduling on parallel machines. Journal of Scheduling, 1(1):55–66, 1998.

[3] R. Askey. Orthogonal polynomials and special functions, volume 21. Siam, 1975.

[4] C. Bachoc and F. Vallentin. New upper bounds for kissing numbers from semidefinite programming. Journal of the American Mathematical Society, 21(3):909–924, 2008.

[5] Y. Bai, E. de Klerk, D. Pasechnik, and R. Sotirov. Exploiting group symmetry in truss topology optimization. Optimization and Engineering, 10(3):331–349, 2009.

[6] B. Barak, S. O. Chan, and P. K. Kothari. Sum of squares lower bounds from pairwise independence. In Proceedings of the forty-seventh annual ACM symposium on Theory of computing, pages 97–106. ACM, 2015.

[7] B. Barak, S. B. Hopkins, J. Kelner, P. Kothari, A. Moitra, and A. Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. In Foundations of Computer Science (FOCS), pages 428–437. IEEE, 2016.

[8] B. Barak and A. Moitra. Noisy tensor completion via the sum-of-squares hierarchy. In Conference on Learning Theory, pages 417–445, 2016.

[9] B. Barak and D. Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. arXiv preprint arXiv:1404.5236, 2014.

[10] G. Blekherman, J. Gouveia, and J. Pfeiffer. Sums of squares on the hypercube. Mathematische Zeitschrift, 284(1-2):41–54, 2016.

[11] E. Chlamtac and M. Tulsiani. Convex relaxations and integrality gaps. In Handbook on semidefinite, conic and polynomial optimization, pages 139–169. Springer, 2012.

[12] E. de Klerk, J. Maharry, D. V. Pasechnik, R. B. Richter, and G. Salazar. Improved bounds for the crossing numbers of $km, n$ and $kn$. SIAM Journal on Discrete Mathematics, 20(1):189–202, 2006.

[13] E. De Klerk, D. V. Pasechnik, and A. Schrijver. Reduction of symmetric semidefinite programs using the regular $*$-representation. Mathematical programming, 109(2-3):613–624, 2007.
[14] M. Dostert, C. Guzmán, F. M. de Oliveira Filho, and F. Vallentin. New upper bounds for the
density of translative packings of three-dimensional convex bodies with tetrahedral symme-
try. *Discrete & Computational Geometry*, 58(2):449–481, 2017.

[15] F. Eisenbrand and R. Weismantel. Proximity results and faster algorithms for integer pro-
gramming using the steinitz lemma. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM
Symposium on Discrete Algorithms*, SODA 2018, pages 808–816, 2018.

[16] K. Gatermann and P. A. Parrilo. Symmetry groups, semidefinite programs, and sums of
quares. *Journal of Pure and Applied Algebra*, 192(1-3):95–128, 2004.

[17] D. Gijswijt, A. Schrijver, and H. Tanaka. New upper bounds for nonbinary codes based on
the terwilliger algebra and semidefinite programming. *Journal of Combinatorial Theory, Series
A*, 113(8):1719–1731, 2006.

[18] M. Goemans and T. Rothvoß. Polynomiality for bin packing with a constant number of item
types. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*,
pages 830–839. SIAM, 2014.

[19] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and
satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145,
1995.

[20] R. L. Graham. Bounds for certain multiprocessing anomalies. *The Bell System Technical Journal*,
45(9):1563–1581, Nov 1966.

[21] D. Grigoriev. Complexity of positivstellensatz proofs for the knapsack. *Computational Com-
plexity*, 10(2):139–154, 2001.

[22] D. Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the
parity. *Theoretical Computer Science*, 259(1-2):613–622, 2001.

[23] N. Gvozdenovic and M. Laurent. The operator $\psi$ for the chromatic number of a graph. *SIAM
Journal on Optimization*, 19:572–591, 2008.

[24] N. Gvozdenović, M. Laurent, and F. Vallentin. Block-diagonal semidefinite programming
hierarchies for 0/1 programming. *Operations Research Letters*, 37(1):27–31, 2009.

[25] D. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Co., 1996.

[26] D. Hochbaum and D. Shmoys. Using dual approximation algorithms for scheduling problems
theoretical and practical results. *Journal of the ACM*, 34:144–162, 1987.

[27] S. B. Hopkins, P. K. Kothari, A. Potechin, P. Raghavendra, T. Schramm, and D. Steurer. The
power of sum-of-squares for detecting hidden structures. In *Foundations of Computer Science
(FOCS)*, pages 720–731. IEEE, 2017.

[28] R. Jans. Solving lot-sizing problems on parallel identical machines using symmetry-breaking
constraints. *INFORMS Journal on Computing*, 21(1):123–136, 2009.
[29] K. Jansen. An eptas for scheduling jobs on uniform processors: using an milp relaxation with a constant number of integral variables. *SIAM Journal on Discrete Mathematics*, 24(2):457–485, 2010.

[30] K. Jansen, K. Klein, and J. Verschae. Closing the gap for makespan scheduling via sparsification techniques. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016*, pages 72:1–72:13, 2016.

[31] V. Kaibel, M. Peinhardt, and M. E. Pfetsch. Orbitopal fixing. *Discrete Optimization*, 8:595–610, 2011.

[32] V. Kaibel and M. Pfetsch. Packing and partitioning orbitopes. *Mathematical Programming*, 114(1):1–36, 2008.

[33] A. Karlin, C. Mathieu, and C. Nguyen. Integrality gaps of linear and semi-definite programming relaxations for knapsack. In *IPCO*, pages 301–314. 2011.

[34] P. Kothari, R. O’Donnell, and T. Schramm. Sos lower bounds with hard constraints: think global, act local. *arXiv preprint arXiv:1809.01207*, 2018.

[35] P. K. Kothari, R. Mori, R. O’Donnell, and D. Witmer. Sum of squares lower bounds for refuting any csp. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 132–145. ACM, 2017.

[36] P. K. Kothari, J. Steinhardt, and D. Steurer. Robust moment estimation and improved clustering via sum of squares. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1035–1046. ACM, 2018.

[37] A. Kurpisz, S. Leppänen, and M. Mastrolilli. On the hardest problem formulations for the 0/1 lasserre hierarchy. *Mathematics of Operations Research*, 42(1):135–143, 2016.

[38] A. Kurpisz, S. Leppänen, and M. Mastrolilli. Sum-of-squares hierarchy lower bounds for symmetric formulations. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 362–374. Springer, 2016.

[39] A. Kurpisz, S. Leppänen, and M. Mastrolilli. An unbounded sum-of-squares hierarchy integrality gap for a polynomially solvable problem. *Mathematical Programming*, 166(1-2):1–17, 2017.

[40] A. Kurpisz, M. Mastrolilli, C. Mathieu, T. Mömke, V. Verdugo, and A. Wiese. Semidefinite and linear programming integrality gaps for scheduling identical machines. *Mathematical Programming*, 172(1-2):231–248, 2018.

[41] J. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11:796–817, 2001.

[42] M. Laurent. A comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0-1 programming. *Mathematics of Operations Research*, 28:470–496, 2003.

[43] M. Laurent. Lower bound for the number of iterations in semidefinite hierarchies for the cut polytope. *Mathematics of operations research*, 28(4):871–883, 2003.
[44] M. Laurent. Semidefinite representations for finite varieties. *Mathematical Programming*, 109(1):1–26, 2007.

[45] E. Levey and T. Rothvoss. A lasserre-based (1+ eps)-approximation for pm—pj=1,prec—cmax. STOC, 2016.

[46] L. Liberti. Automatic Generation of Symmetry-Breaking Constraints. In *Combinatorial Optimization and Applications*, Lecture Notes in Computer Science, pages 328–338. Springer, Berlin, Heidelberg, 2008.

[47] L. Liberti. Symmetry in Mathematical Programming. In *Mixed Integer Nonlinear Programming*, The IMA Volumes in Mathematics and its Applications, pages 263–283. Springer, New York, NY, 2012.

[48] B. Litjens, S. Polak, and A. Schrijver. Semidefinite bounds for nonbinary codes based on quadruples. *Designs, Codes and Cryptography*, 84(1-2):87–100, 2017.

[49] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1:166–190, 1991.

[50] T. Ma, J. Shi, and D. Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *Foundations of Computer Science (FOCS)*, pages 438–446. IEEE, 2016.

[51] F. Margot. Pruning by isomorphism in branch-and-cut. *Mathematical Programming*, 94:71–90, 2002.

[52] F. Margot. Symmetry in integer linear programming. In *50 Years of Integer Programming 1958-2008*, pages 647–686. Springer, 2010.

[53] J. Ostrowski. Using symmetry to optimize over the Sherali–Adams relaxation. *Mathematical Programming Computation*, 6:405–428, 2014.

[54] J. Ostrowski, J. Linderoth, F. Rossi, and S. Smriglio. Orbital branching. *Mathematical Programming*, 126:147–178, 2011.

[55] P. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96:293–320, 2003.

[56] A. Potechin. Sum of squares lower bounds from symmetry and a good story. *arXiv preprint arXiv:1711.11469*, 2017.

[57] A. Potechin and D. Steurer. Exact tensor completion with sum-of-squares. *arXiv preprint arXiv:1702.06237*, 2017.

[58] P. Raghavendra, T. Schramm, and D. Steurer. High-dimensional estimation via sum-of-squares proofs. *arXiv preprint arXiv:1807.11419*, 2018.

[59] A. Raymond, J. Saunderson, M. Singh, and R. R. Thomas. Symmetric sums of squares over k-subset hypercubes. *Mathematical Programming*, 167(2):315–354, 2018.

[60] A. A. Razborov. Flag algebras. *The Journal of Symbolic Logic*, 72(4):1239–1282, 2007.
A. A. Razborov. On 3-hypergraphs with forbidden 4-vertex configurations. *SIAM Journal on Discrete Mathematics*, 24(3):946–963, 2010.

C. Riener, T. Theobald, L. J. Andrén, and J. B. Lasserre. Exploiting symmetries in sdp-relaxations for polynomial optimization. *Mathematics of Operations Research*, 38(1):122–141, 2013.

T. Rothvoß. The lasserre hierarchy in approximation algorithms. *Lecture notes for the MAPSP*, 2013.

B. Sagan. The symmetric group, volume 203 of graduate texts in mathematics, 2001.

G. Schoenebeck. Linear level lasserre lower bounds for certain k-csp's. In *Foundations of Computer Science (FOCS)*, pages 593–602. IEEE, 2008.

H. Sherali and W. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3:411–430, 1990.

O. Svensson. Santa claus schedules jobs on unrelated machines. *SIAM Journal on Computing*, 41(5):1318–1341, 2012.

D. Williamson and D. Shmoys. *The design of approximation algorithms*. Cambridge University Press, 2011.

5 Appendix

We show how to prove Theorem 4 following the lines in the work of Raymond et al. [59]. We need a few intermediate results, and the symmetry reduction theorem from Gatermann & Parrillo, stated in our setting [16].

**Theorem 11 ([16]).** Suppose that \(g \in \mathbb{R}[y]/I_E\) is a degree-\(\ell\) SoS and \(S_m\)-invariant polynomial. For each partition \(\lambda \vdash m\), let \(\tau_\lambda\) be a tableau of shape \(\lambda\) and let \(\{b_1^\lambda, \ldots, b_{m_\lambda}^\lambda\}\) be a basis \(W_{\tau_\lambda}\). Then, for each partition \(\lambda \vdash m\) there exists a \(m_\lambda \times m_\lambda\) positive semidefinite matrix \(Q_\lambda\) such that \(g = \sum_{\lambda \vdash m} \langle Q_\lambda, Y^\lambda \rangle\), where \(Y^\lambda_{ij} = \text{sym}(b_i^\lambda b_j^\lambda)\).

Given two partitions \(\lambda, \mu\), we say that \(\lambda \geq \mu\) is \(\lambda \geq_{\text{lex}} \mu\) and the number of parts of \(\mu\) is at least the number of parts of \(\lambda\). The following is a consequence of Young’s rule, and we refer to [59, Lemma 1] for a proof.

**Lemma 17.** Let \(V\) be a finite-dimensional \(S_m\)-module. If \(\tau_\mu\) has shape \(\mu\), then \(V_{R_{\tau_\mu}} \subseteq \bigoplus_{\lambda \geq_{\text{lex}} \mu} V_\lambda\).

The following lemma is a variant of [59, Theorem 2] for the action of the symmetric group in our setting. Together with previous lemma and the theorem of Gatermann & Parrillo we can conclude Theorem 4.

**Lemma 18.** The dimension \(m_\lambda\) of \(Q_\lambda^\ell\) in the isotypic decomposition of \(Q^\ell\) is zero unless \(\lambda \geq_{\text{lex}} (m - \ell, 1^\ell)\).
Proof. Let $x_S$ be a monomial of degree at most $\ell$ with $S = \{(i_k, C_k) : k \in [\ell]\}$. In particular, $|\{i_k : k \in [\ell]\}| \leq \ell$. Let $\tau$ be any tableau with shape $(m - \ell, 1^\ell)$, where the tail of $\tau$ contains every elements of $\{i_k : k \in [\ell]\}$. The subgroup $R_\tau$ fixes $S$, therefore $x_S \in W_\tau$, and we have then

$$Q_\ell \subseteq \bigoplus_{\tau : \text{shape}(\tau) = (m-\ell,1^\ell)} W_\tau \subseteq \bigoplus_{\lambda \geq (m-\ell,1^\ell)} Q_\lambda'.$$

To conclude, observe that if $\lambda \geq (m - \ell, 1^\ell)$ then $\lambda_1 \geq m - \ell$. Since $\lambda \vdash m$, the maximum number of parts for $\lambda$ is $m - \lambda_1 \leq \ell$, that is, $\lambda$ has at most $\ell + 1$ parts. Therefore, $\lambda \geq (m - \ell, 1^\ell)$ if and only if $\lambda \geq_{\text{lex}} (m - \ell, 1^\ell)$.

Proof of Theorem 4. Let $g \in \mathbb{R}[y]/I_E$ be a degree-$\ell$ SoS and $S_m$-invariant polynomial. By Theorem 11 and Lemma 18, for each $\lambda \in \Lambda_\ell$ there exists a positive semidefinite matrix $Y^\lambda$ such that $g = \sum_{\lambda \in \Lambda_\ell} \langle Q_\lambda, Y^\lambda \rangle$. Since $\{b_1^\lambda, \ldots, b_{m_\lambda}^\lambda\} \subseteq \text{span}(P^\lambda)$, there exists a real matrix $T_\lambda$ such that

$$T_\lambda(p_1^\lambda, \ldots, p_{m_\lambda}^\lambda) = (b_1^\lambda, \ldots, b_{m_\lambda}^\lambda).$$

Consider the congruent transformation $M_\lambda = T_\lambda^\top Q_\lambda T_\lambda$. In particular, $M_\lambda$ is also positive semidefinite. Furthermore,

$$b^\top Q_\lambda b = (T_\lambda p)^\top Q_\lambda (T_\lambda p) = p^\top M_\lambda p,$$

where $b = (b_1^\lambda, \ldots, b_{m_\lambda}^\lambda)$ and $p = (p_1^\lambda, \ldots, p_{m_\lambda}^\lambda)$. That is, $g = \sum_{\lambda \in \Lambda_\ell} \langle Q_\lambda, Y^\lambda \rangle = \sum_{\lambda \in \Lambda_\ell} \langle M_\lambda, Z^\lambda \rangle$. \qed