Using Structural Properties for Integer Programs

Sebastian Berndt[1] and Kim-Manuel Klein[2]

[1] Department of Computer Science, Kiel University
seb@informatik.uni-kiel.de
[2] École Polytechnique Fédérale de Lausanne
kim-manuel.klein@epfl.ch

Abstract. Integer programs (IPs) are one of the fundamental tools used to solve combinatorial problems in theory and practice. Understanding the structure of solutions of IPs is thus helpful to argue about the existence of solutions with a certain simple structure, leading to significant algorithmic improvements. Typical examples for such structural properties are solutions that use a specific type of variable very often or solutions that only contain few non-zero variables. The last decade has shown the usefulness of this method. In this paper we summarize recent progress for structural properties and their algorithmic implications in the area of approximation algorithms and fixed parameter tractability. Concretely, we show how these structural properties lead to optimal approximation algorithms for the classical Makespan Scheduling scheduling problem and to exact polynomial-time algorithm for the Bin Packing problem with a constant number of different item sizes.

1 Introduction

Integer programming is one of the fundamental tools used to solve scheduling problems in practice. Understanding the structure of optimal solutions of integer programs is helpful in many ways: First, we might want to find optimal solutions that have certain characteristics like having a small number of non-zero variables. Using structural theorems, we can show the (non)-existence of solutions with such characteristics. Second, understanding the structure of the integer program might also lead to algorithmic improvements: If one knows that some optimal solutions belong to a certain set S, we only need to search through S instead of the whole solution space.

Our main focus is on IPs of the following form. Given a polytope $\mathcal{P} = \{x \in \mathbb{R}^d \mid Ax \leq b\}$ for some matrix $A \in \mathbb{Z}^{d \times m}$ and vector $b \in \mathbb{Z}^d$. Then the IP is defined by

$$\begin{aligned} &\min \sum x_p \\ &\sum_{p \in \mathcal{P} \cap \mathbb{Z}^d} x_p p = a \qquad (1) \\ &x \in \mathbb{Z}^d_{\geq 0}, \end{aligned}$$

with variables $x = (x_p)_{p \in \mathcal{P} \cap \mathbb{Z}^d}$. Throughout this work, let α be the largest entry in $\mathcal{P} \cap \mathbb{Z}^d$, i. e. $\alpha = \max_{p \in \mathcal{P} \cap \mathbb{Z}^d}\{||p||_\infty\}$. IPs of this specific shape are especially interesting when it comes to packing and scheduling problems, as the well-known *configuration IPs* are of this form. Intuitively, a configuration corresponds to a certain schedule of a single machine or a valid packing of a single object such as a bin. The configuration IP then chooses suitable configurations depending on the concrete programs. Here, we mainly consider the classical Bin Packing problem and the classical Makespan Scheduling problem on identical machines. In this paper, we summarize and explain the structural properties needed to obtain two important algorithmical advances in combinatorial optimization of the recent years.

Bin Packing*:* In the Bin Packing problem, item sizes $s_1, \ldots, s_d \in (0,1]$ are given. Every item size s_i occurs with a certain multiplicity a_i and the objective is to pack the given item into as few unit sized bins as possible. This problem can be written in the form of IP (1) by setting $\mathcal{P} = \{x \in \mathbb{R}^d_{\geq 0} \mid s_1x_1 + \ldots + s_dx_d \leq 1\}$. This polytope $\mathcal{P}$ is called the *knapsack polytope* and an element $p \in \mathcal{P}$ (called a *pattern*) corresponds to a valid packing of a single bin. Hence, the optimal solution of IP (1) corresponds to a packing with a minimal number of bins.

For the Bin Packing problem we give an overview of the structural properties needed to obtain the following theorem, which solved a long standing open problem (see e. g. [3, 4, 10]).

Theorem 1 (Goemans, Rothvoß [5]). *Assuming the number of different item sizes d is constant, then there exists a polynomial time algorithm for the* Bin Packing *problem.*

Makespan Scheduling*:* In the Makespan Scheduling problem on identical machines, a set of n jobs with processing times $p_1, \ldots, p_d$ are given. The goal is to assign all jobs to m machines with identical speeds such that the maximum load over all machines (the so called *makespan*) is minimized. If T is a guess for the optimal makespan, we can determine whether a schedule with makespan at most T exists by using IP (1): We set $\mathcal{P}_T = \{x \in \mathbb{R}^d_{\geq 0} \mid p_1x_1 + \ldots + p_dx_d \leq T\}$ as the corresponding knapsack polytope. An element $p \in \mathcal{P}_T$ (called a *configuration*) corresponds to the packing of a single machine. Any solution of IP (1) with objective value at most m thus corresponds to a schedule with makespan at most T.

For the Makespan Scheduling problem, we summarize an algorithmic result in the area of approximation algorithms that settles the complexity of this problem (under a widely believed complexity assumption):

Theorem 2 (Jansen, Klein, Verschae [7]). *There is an approximation algorithm for the classical* Makespan Scheduling *problem on identical machines with a running time of*

$$2^{O(1/\epsilon \cdot \log^4(1/\epsilon))} + O(n \log n),$$

that produces an $(1+\epsilon)$-approximation.

The optimality of this algorithm is based on a theorem due to Chen et al. [2], which states that for all $\delta > 0$, the existence of an $(1+\epsilon)$-approximation algorithm for the Makespan Scheduling problem on identical machines running in time $2^{O([1/\epsilon]^{1-\delta})} \cdot \mathrm{poly}(n)$ implies that the *Exponential Time Hypothesis (ETH)* is false. The ETH states that the satisfiability problem SAT can not be solved in truly subexponential time.

As we will see, both results are based on elementary structural results of the IP (1). However, the structural properties themselves differ in a very important point. In case of the scheduling problem, we do not require an optimal solution. This allows us to round processing times and ignore small jobs, as those small jobs can be placed greedily on an existing schedule without losing too much. Hence the entries of the Makespan Scheduling IP turn out to be relatively small (smaller than $1/\epsilon$). Therefore, bounds in the structural properties may depend on the largest entry in the IP. In the Bin Packing problem however, where the goal is to get an optimum solution, small items can not be added greedily and therefore have to be placed optimally by the IP. This implies that the sizes of entries (which correspond to the inverse of the sizes of the items) of the IP can be very large. Therefore, structural properties are needed, that yield bounds independent of the size of the entries in IP (1).

2 Solutions of Bounded Support

One of the first structural results for IP (1) was to show that there always exists a solution (assuming the IP is feasible) with a bounded support. The support $\mathrm{supp}(\lambda)$ of a solution $\lambda \in \mathbb{Z}^d$ is defined to be the set of non-zero components, i. e. $\mathrm{supp}(\lambda) = \{i \mid \lambda_i > 0\}$ for $\lambda = (\lambda_1, \ldots, \lambda_d)$.

Theorem 3 (Eisenbrand, Shmonin [3]). *Assuming that IP* (1) *is feasible, then there exists a feasible solution $\lambda \in \mathbb{Z}^d$ of* (1) *with*

$$|\,\mathrm{supp}(\lambda)| \leq 2d \cdot \log(4d\alpha).$$

Actually this result does not only hold for IP (1) but for arbitrary integer programs with entries bounded by α. Very recently, this bound has been improved by Aliev et al. [1] to $|\,\mathrm{supp}(\lambda)| \leq 2d \cdot \log(2\sqrt{d}\alpha)$. Furthermore, they showed that this bound also holds for *optimal solutions* and arbitrary objective functions $\min c^T x$.

Similar to the previous theorem, Eisenbrand and Shmonin [3] proved a bound for the support of an integral solution of IP (1) that does not depend on the largest entry α in $\mathcal{P} \cap \mathbb{Z}^d$. We also give a sketch of the proof due to its very elementary nature.

Theorem 4 (Eisenbrand, Shmonin [3]). *Assuming IP* (1) *is feasible, then there exists a feasible solution λ of* (1) *such that $|supp(\lambda)| \leq 2^d$.*

Proof. We assign a *potential* $\varphi(\lambda)$ to each feasible solution λ of IP (1), with

$$\varphi(\lambda) = \sum_{p \in \mathcal{P} \cap \mathbb{Z}^d} \lambda_p \, || \, (1, p)^\top \, ||,$$

where $|| \cdot ||$ is the Euclidean norm in $\mathbb{R}^{d+1}$. Let λ^* be the solution of IP (1) with the smallest potential. If $|\operatorname{supp}(\lambda^*)| > 2^d$, the pidgeonhole principle implies that there are $p, p' \in \mathcal{P} \cap \mathbb{Z}^d$ with same parity, i. e. $p \equiv p' \bmod 2$. Hence, $\hat{p} = (p+p')/2$ is also an element of $\mathcal{P} \cap \mathbb{Z}^d$. Reassigning weight from p and p' to $\hat{p}$ thus gives us a new feasible solution $\hat{\lambda}$ with $\varphi(\hat{\lambda}) < \varphi(\lambda^*)$, as $(1, p)^\top$ and $(1, p')^\top$ are not co-linear. This is a contradiction to the minimality of $\varphi(\lambda^*)$. Hence, the solution with minimal potential has support of size at most 2^d. $\square$

3 Structural Results for IPs with Large Entries

By the previous section, we got an understanding of the support of solutions of IP (1). However, we would actually like to know more about the shape of these solutions. For example, we would like to understand what kind of configurations are actually used and what their respective multiplicities are. In this spirit, Goemans and Rothvoß showed the following theorem in order to prove Theorem 1.

Theorem 5 (Goemans, Rothvoß [5]). *Let $\mathcal{P}$ be a polytope as in IP (1). There exists a distinguished set of configurations $X \subseteq \mathcal{P} \cap \mathbb{Z}^d$ with $|X| \leq m^d d^{O(d)} (\log \Delta)^d$ such that for every right hand side a of IP (1), there exists a vector $\lambda \in \mathbb{Z}_{\geq 0}^{\mathcal{P} \cap \mathbb{Z}^d}$ with $\sum_{p \in \mathcal{P} \cap \mathbb{Z}^d} \lambda_p p = a$ and*

1. $\lambda_p \leq 1 \qquad \forall p \in (\mathcal{P} \cap \mathbb{Z}^d) \setminus X$
2. $|supp(\lambda) \cap X| \leq 2^{2d}$
3. $|supp(\lambda) \setminus X| \leq 2^{2d}$

Furthermore, X can be computed in time $\operatorname{poly}(|X|)$.

The set $X \subseteq \mathcal{P} \cap \mathbb{Z}^d$ of distinguished configurations actually comes from a rather technical covering of the polytope $\mathcal{P}$ into parallelepipedes. The set X is then the set of vertices of the parallelepipedes. So one might wonder, if there exists a structure theorem with a rather natural set of extinguished point. Therefore, we consider the integer hull $\mathcal{P}_I$ of the polytope $\mathcal{P}$ which is defined by the convex hull of all integral points inside $\mathcal{P}$, i. e. $\mathcal{P}_I = Conv(\mathcal{P} \cap \mathbb{Z}^d)$. The following structure theorem uses the set of vertices V_I of $\mathcal{P}_I$, which gives a more natural alternative to the set X used in Theorem 5.

Theorem 6 (Jansen, Klein [6]). *Let $\mathcal{P}$ be a polytope as in IP (1). For every right hand side a of IP (1), there exists a vector $\lambda \in \mathbb{Z}_{\geq 0}^{\mathcal{P} \cap \mathbb{Z}^d}$ with $\sum_{p \in \mathcal{P} \cap \mathbb{Z}^d} \lambda_p p = a$ and*

1. $\lambda_p \leq 2^{2^{O(d)}} \quad \forall p \in (\mathcal{P} \cap \mathbb{Z}^d) \setminus V_I$

2. $|supp(\lambda) \cap V_I| \leq d \cdot 2^d$
3. $|supp(\lambda) \setminus V_I| \leq 2^{2d}$

Theorem 6 can now be used to prove Theorem 1. Note that the cardinality of the set of vertices V_I of $\mathcal{P}_I$ is bounded by $d^d \cdot O(\log \alpha)^d$ and can be computed in time $|V_I| \cdot d^{O(d)}$. The right hand side a is given by the Bin Packing instance. In order to determine the solution λ implied by Theorem 6, we first find the set $V_\lambda = \text{supp}(\lambda) \cap V_I$. Due to the second property of λ, there are at most $\binom{|V_I|}{d \cdot 2^d} \leq |V_i|^{O(2^d)}$ such sets. We also guess the cardinality of the set $\bar{V}_\lambda = \text{supp}(\lambda) \setminus V_I$. Due to the third property of λ, we have $|\bar{V}_\lambda| \leq 2^{2d}$ and there are thus at most 2^{2d} choices for k.

For guess V_λ and guess $|\bar{V}_\lambda|$, we need to (i) find the multiplicities λ_p for the pattern $p \in V_\lambda$ and we need to (ii) find the elements of $\bar{V}_\lambda$ and their multiplicities. For (i), let $\hat{\lambda}_p$ be a variable describing the multiplicity of $p \in V_\lambda$. In order to solve (ii), the i-th element of $\bar{V}_\lambda$ will be described by d variables $x_{i,1}, x_{i,2}, \ldots, x_{i,d}$. As we also need to determine the multiplicity of the i-th element of $\bar{V}_\lambda$, we actually use variables $x_{i,j}^{(\ell)}$ indicating the j-th coordinate of the i-th element with multiplicity 2^ℓ. Note that there are at most 2^d variables $\hat{\lambda}_p$ and at most $d \cdot 2^{O(d)}$ variables $x_{i,j}^\ell$. One can thus formulate this task as an integer program with at most $2^{O(d)}$ variables which can be solved in time $(2^{O(d)})^{O(d)} \cdot \text{poly}(n)$ by the algorithms of Lenstra and Kannan [8, 9].

By using a binary search on the number of bins, we can thus solve the Bin Packing problem in time $\text{poly}(n)^{f(d)}$ for some function f, which proves Theorem 1. A closer inspection shows that the running time is actually of the form $f(|V_I|) \cdot \text{poly}(n)$, the Bin Packing problem is thus *fixed parameter tractable* for parameter $|V_I|$. See [6] for a more detailed discussion on this.

4 Structural Results for IPs with Small Entries

In contrast to the previous section, we are now interested in IPs that only contain small numbers. In order to determine the optimal makespan, we perform a binary search. Let T be the current guess and $\mathcal{P}_T = \{x \in \mathbb{R}^d_{\geq 0} \mid p_1 x_1 + \ldots + p_d x_d \leq T\}$ be the polytope of valid configurations. To simplify notation, let π be the vector of the processing times. A configuration $p \in \mathcal{P}_T$ is called *simple* if $|\text{supp}(p)| \leq \log(T+1)$. Otherwise, it is called *complex*. Let P_C be the set of complex configurations and P_S be the set of simple configurations. In the Bin Packing setting, we saw that there is always an optimal solution that is mostly built on elements of V_I. Similarly, we will show that the Makespan Scheduling problem always has an optimal solution that is mostly build on the simple configurations P_S.

Theorem 7 (Jansen, Klein, Verschae [7]). *Let α be the largest entry of any vector in $\mathcal{P}_T \cap \mathbb{Z}^d$. Assume that IP* (1) *is feasible. Then there exists a feasible solution λ to* (1) *such that:*

1. if $\lambda_p > 1$ then the configuration p is simple,

2. *the support of* λ *satisfies* $|\operatorname{supp}(\lambda)| \leq 4d\log(4d\alpha)$*, and*
3. $\sum_{p\in P_C} \lambda_p \leq 2d\log(4d\lambda)$*, where* P_C *denotes the set of complex configurations.*

Proof. The main idea in the proof is that for any complex configuration p, we can write $2p$ as $2p = p_1 + p_2$ for two configurations p_1, p_2 with $\pi \cdot p_1 = \pi \cdot p_2$ and $\operatorname{supp}(p_i) \subsetneq \operatorname{supp}(p)$. For $S \subseteq \operatorname{supp}(p)$, let $p[S]$ be the configuration with

$$p[S]_i = \begin{cases} 1 & i \in S \\ 0 & \text{else.} \end{cases}$$

Note that $|\{p[S] \mid S \subseteq \operatorname{supp}(p)\}| = 2^{|\operatorname{supp}(p)|} > 2^{\log(T+1)} = T+1$. As $\pi \cdot p[S] \leq \pi \cdot p \leq T$, the pidgeonhole principle implies that there are different sets $S_1, S_2 \subseteq \operatorname{supp}(p)$ with $\pi \cdot p[S_1] = \pi \cdot p[S_2]$. Clearly, $\pi \cdot p[S_1'] = \pi \cdot p[S_2']$ also holds for $S_1' = S_1 \setminus S_2$ and $S_2' = S_2 \setminus S_1$. Then $p_1 = p + p[S_1'] - p[S_2']$ and $p_2 = p + p[S_2'] - p[S_1']$ are the desired configurations.

If $\lambda^{(0)}$ is a feasible solution, we can now iteratively reduce complex configurations $p \in P_C$ with $\lambda_p^{(0)} \geq 2$ into smaller configurations p_1 and p_2 and achieve a solution $\lambda^{(1)}$. We repeat this, until our final solution $\lambda^{(k)}$ has the property that $\lambda_p^{(k)} \leq 1$ for all $p \in P_C$. Finally, we can use Theorem 3 to bound the cardinality of the support. One needs to be careful here, as the solution implied by Theorem 3 might contain a complex configuration with higher multiplicity. This problem can be avoided by making use of a potential function that guarantees that complex configurations do not reappear in this step. □

If one wants to compute a $(1+\epsilon)$-approximation, we can discard all processing times smaller than $\epsilon \cdot \mathrm{LB} = \epsilon \cdot \max\{p_{\max}, \sum_j p_j/m\}$, where $p_{\max}$ is the maximal processing time of a single job. The discarded jobs can be added greedily later on. This process is the reason that the corresponding IP has only small entries. The remaining n' large jobs with processing time p such that $\epsilon\,\mathrm{LB} \leq p \leq \mathrm{LB}$ can be rounded geometrically to the form $\epsilon \cdot \mathrm{LB} \cdot (1+\epsilon)^i$ without losing too much precision. After this preprocessing, the number of different processing times is thus $d' = O(1/\epsilon \log(1/\epsilon))$. By normalizing our instance, we can assume that $T \leq 1/\epsilon^2$ and that all processing times are integers between $1/\epsilon$ and $1/\epsilon^2$. Let $\pi' = (\pi_1', \ldots, \pi_{d'}')$ be the vector corresponding to these processing times and n_j' be the number of jobs with processing times π_j'.

By formulating IP (1) for this reduced problem, we can use the following approach:

1. For each $j = 1, \ldots, d'$, guess the number $n_j^C \leq n_j'$ of jobs covered by complex configurations.
2. Find the minimum number of machines m^C to schedule the $n^C = \sum_j n_j^C$ jobs with makespan T.
3. To schedule the remaining jobs with simple configurations, guess the simple configurations $S \subseteq P_S$.
4. Solve the IP (1) restricted to the $n^S = n' - n'^C$ jobs that will be scheduled by simple configurations in S.

For step 1, note that Theorem 7 guarantees that m^C – the number machines scheduled according to complex configurations – is at most $2d' \log(4d'\alpha) \in O(1/\epsilon \log(1/\epsilon))$. As only $1/\epsilon$ jobs belong to a configuration (they are of size at least $1/\epsilon$), there are at most $O(1/\epsilon^2 \log(1/\epsilon))$ many jobs covered by complex configurations. There are thus at most $\sum_{k=0}^{O(1/\epsilon^2 \log(1/\epsilon))} \binom{k+d'-1}{d'-1} \leq 2^{O(1/\epsilon \log^2(1/\epsilon))}$ many such choices. Step 2 can thus be solved by a dynamic program in the same running time of $2^{O(1/\epsilon \log^2(1/\epsilon))}$. For step 3, note that $|P_S| \leq 2^{O(\log^2(1/\epsilon))}$, as $T \leq 1/\epsilon^2$ and each job has size at least $1/\epsilon$. As our desired solution uses at most $4d' \log(4d'\alpha) \in O(1/\epsilon \log^2(1/\epsilon))$ different configurations in total, we can try out all $\sum_{k=0}^{O(1/\epsilon \log^2(1/\epsilon))} \binom{|P_S|}{k} \leq 2^{O(1/\epsilon \log^4(1/\epsilon))}$ choices for the support S of the simple configurations. Finally, the IP in step 4 has at most $O(1/\epsilon \log^2(1/\epsilon))$ many variables and can thus be solved by the algorithms of Lenstra and Kannan [8, 9] in time $2^{O(1/\epsilon \log^3(1/\epsilon))} \cdot \log(n)$. Due to the preprocessing and the binary search, the complete algorithm thus runs in time $2^{O(1/\epsilon \log^4(1/\epsilon))} + O(n \log n)$. This proves Theorem 2.

5 Conclusions

Understanding the structure of solutions of combinatorial problems has long been one of the most successful strategies to design efficient algorithms. Recent advances in the theory of integer programming allow to take a more closely look at the structure of solution space of IP formulations of these combinatorial problems. This led to a series of exciting results for longstanding open problems. We have seen that Bin Packing can be solved in time $f(|V_I|) \cdot \mathrm{poly}(n)$, which can be bounded by $\mathrm{poly}(n)^{g(d)}$ for some function g. Hence, for constant d, this running time is polynomial. A major open question is whether one can obtain a running time of $f(d) \cdot \mathrm{poly}(n)$, i. e. whether the Bin Packing problem is fixed parameter tractable with parameter d.

References

[1] I. Aliev, J. De Loera, F. Eisenbrand, T. Oertel, and R. Weismantel. “The Support of Integer Optimal Solutions”. In: *ArXiv* (2017). 1712.08923.

[2] L. Chen, K. Jansen, and G. Zhang. “On the optimality of approximation schemes for the classical scheduling problem”. In: *Proc. SODA*. SIAM, 2014, pp. 657–668.

[3] F. Eisenbrand and G. Shmonin. “Carathéodory bounds for integer cones”. In: *Operations Research Letters* 34.5 (2006), pp. 564–568.

[4] C. Filippi. “On the bin packing problem with a fixed number of object weights”. In: *European Journal of Operational Research (EJOR)* 181.1 (2007), pp. 117–126.

[5] M. X. Goemans and T. Rothvoß. “Polynomiality for Bin Packing with a Constant Number of Item Types”. In: *Proc. SODA*. 2014, pp. 830–839.

[6] K. Jansen and K. Klein. “About the Structure of the Integer Cone and its Application to Bin Packing”. In: *Proc. SODA*. SIAM, 2017, pp. 1571–1581.

[7] K. Jansen, K. Klein, and J. Verschae. “Closing the Gap for Makespan Scheduling via Sparsification Techniques”. In: *Proc. ICALP*. Vol. 55. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, 72:1–72:13.

[8] R. Kannan. “Minkowski's Convex Body Theorem and Integer Programming”. In: *Mathematics of Operations Research* 12.3 (1987), pp. 415–440.

[9] H. W. Lenstra. “Integer Programming with a Fixed Number of Variables”. In: *Mathematics of Operations Research* 8 (1983), pp. 538–548.

[10] S. T. McCormick, S. R. Smallwood, and F. C. R. Spieksma. “Polynomial Algorithms for Multiprocessor Scheduling with a Small Number of Job Lengths”. In: *Proc. SODA*. SIAM, 1997, pp. 509–517.