Journal of Computer and System Sciences 82 (2016) 758–766

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

On polynomial kernels for sparse integer linear programs

Stefan Kratsch [1]

Department for Computer Science, University of Bonn, Germany

ARTICLE INFO

Article history:
Received 29 January 2015
Received in revised form 29 September 2015
Accepted 19 December 2015
Available online 12 January 2016

Keywords:
Parameterized complexity
Kernelization
Integer linear programs

ABSTRACT

Successful solvers for integer linear programs (ILPs) demonstrate that preprocessing can greatly speed up the computation. We study preprocessing for ILPs via the theoretical notion of kernelization from parameterized complexity. Prior to our work, there were only implied lower bounds from other problems that hold only for dense instances and do not take into account the domain size. We consider the feasibility problem for ILPs $Ax \le b$ where A is an r-row-sparse matrix parameterized by the number of variables, i.e., A has at most r nonzero entries per row, and show that its kernelizability depends strongly on the domain size. If the domain is unbounded then this problem does not admit a polynomial kernelization unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. If, on the other hand, the domain size d of each variable is polynomially bounded in n, or if d is an additional parameter, then we do get a polynomial kernelization.

1. Introduction

The present work seeks to initiate a study of the preprocessing properties of integer linear programs (ILPs) within the framework of parameterized complexity. Generally, preprocessing (or data reduction) is a universal strategy for coping with combinatorially hard problems and can be combined with other strategies like approximation, brute-force, exact exponential-time algorithms, local search, or heuristics. Unlike those other approaches, preprocessing itself incurs only a polynomial-time cost and is error free (or, in rare cases, has negligible error); recall that under standard assumptions we do not expect to exactly solve any NP-hard problem in polynomial time. Thus, preprocessing before applying other paradigms is essentially free and may reduce both error and running time by handling parts of the input that are sufficiently easy in polynomial time (see e.g. [29]). For a long time, preprocessing has been neglected in theoretical research for lack of appropriate tools and research was limited to experimental evaluation of preprocessing strategies. The introduction of parameterized complexity and its notion of kernelization has sparked a strong interest in theoretically studying preprocessing with proven upper and lower bounds on its performance.

Integer linear programs are widely applied in theory and practice. There is a huge body of scientific literature on ILPs both as a topic of research itself and as a tool for solving other problems (see, e.g., [26,28,16]). From a theoretical perspective, many fundamental problems that revolve around ILPs are hard, e.g., checking feasibility of a 0/1-ILP is NP-hard by an easy reduction from the classic Satisfiability problem [14]. Similarly, it is easy to express Vertex Cover or Independent Set, thus showing that simple covering and packing ILPs are NP-hard to optimize. Thus, for worst-case complexity considerations, the

E-mail address: kratsch@cs.uni-bonn.de.

[1] Work done in part at Utrecht University, the Netherlands, supported by the Netherlands Organization for Scientific Research (NWO), project "KERNELS", and in part at Max-Planck-Institute for Informatics, Saarbrücken, Germany.

http://dx.doi.org/10.1016/j.jcss.2015.12.002

high expressive power of ILPs comes at the price of encompassing plenty of hard problems and, effectively, inheriting all their lower bounds (e.g., approximability).

In practice, the expressive power of ILPs makes them a versatile framework for encoding and solving many combinatorially hard problems. Coupled with powerful software packages for optimizing ILPs this has created a viable way for solving many practical problems on real-world instances. We refer to a survey of Atamtürk and Savelsbergh [1] for an explanation of the capabilities of modern ILP solvers; this includes techniques such as probing and coefficient reduction. One of the most well-known solvers is the CPLEX package, which is, in particular, known for its extensive preprocessing options.[2] It is known that appropriate preprocessing and simplification of ILPs can lead to strong improvements in running time, e.g., reducing the domain of variables or eliminating them altogether, or reducing the number of constraints. Given the large number of options that a user has for controlling the preprocessing in CPLEX, e.g., the number of substitution rounds to reduce rows and columns, this involves some amount of engineering and has a more heuristic flavor. In particular, there are no performance guarantees for the effect of the preprocessing.

Naturally, this leads to the question of whether there are theoretical performance guarantees for the viability of preprocessing for ILPs. To pursue this question in a rigorous and formal way, we take the perspective of parameterized complexity and its notion of (polynomial) kernelization. Parameterized complexity studies classical problems in a more fine-grained way by introducing one or more additional parameters and analyzing time- and space-usage as functions of input size and parameter. In particular, by formalizing a notion of *fixed-parameter tractability*, which requires efficient algorithms when the parameter is small, this makes the parameter a quantitative indicator of the hardness of a given instance (see Section 2 for formal definitions). This in turn permits us to formalize preprocessing as a reduction to an equivalent instance of size bounded in the parameter, a so-called *kernelization*. The intuition is that relatively easy instances should be reducible to a computationally hard, but small core, and we do not expect to reduce instances that are already fairly hard compared to their size (e.g., instances that are already reduced). While classically, no efficient algorithm can shrink *each* instance of an NP-hard problem [15], the notion of kernelization has been successfully applied to a multitude of problems (see recent surveys by Lokshtanov et al. [25] and Kratsch [21]). Due to many interesting upper bound results (e.g., [4,12,23]) but also the fairly recent development of a lower bound framework for polynomial kernels [15,13,3,8], the existence or non-existence of polynomial kernels (which reduce to size polynomial in the parameter) is receiving high interest.

In this work, we focus on the effect that the dimension, i.e., the number of variables, has on the preprocessing properties of ILPs. Feasibility and optimization of ILPs with low dimension has been studied extensively already, see e.g. [18,17,24,27,19,6,10,11]. The most important result for our purpose is a well-known work of Lenstra [24], who showed that feasibility of an ILP with n variables and m constraints can be decided in time $\mathcal{O}(c^{n^3} \cdot m^{\mathcal{O}(1)})$; this also means that the problem is fixed-parameter tractable with respect to n. This has been improved further, amongst others by Kannan [19] to $\mathcal{O}(n^{cn})$ dependence on the dimension and by Clarkson [6] to (expected) $\mathcal{O}((cn)^{n/2+\mathcal{O}(1)})$ dependence. We take these results as our starting point and consider the problem of determining feasibility of a given ILP parameterized by the number of variables, formally defined as follows.

INTEGER LINEAR PROGRAM FEASIBILITY(n) — ILPF(n)
Input: A matrix $A \in \mathbb{Z}^{m\times n}$ and a vector $b \in \mathbb{Z}^m$.
Parameter: n.
Output: Is there a vector $x \in \mathbb{Z}^n$ such that $Ax \le b$?

It is known by a simple folklore argument that any parameterized problem is fixed-parameter tractable if and only if it admits a kernelization (cf. [7]); unfortunately the implied size guarantee is usually impractical as it is exponential in the parameter. As an example, using the running time given by Kannan [19] we only get a kernel size of $\mathcal{O}(n^{cn})$.[3] Unsurprisingly, we are more interested in what kernel sizes can be achieved by nontrivial preprocessing rules. In particular, we are interested in the conditions under which an ILP with n variables can be reduced to size polynomial in n, i.e., in the existence of polynomial kernels for INTEGER LINEAR PROGRAM FEASIBILITY(n).

Related work. Regarding the existence of polynomial kernels for INTEGER LINEAR PROGRAM FEASIBILITY(n) only little is known. In general, parameterized by the number of variables, ILPF(n) admits no polynomial kernelization unless NP ⊆ coNP/poly and the polynomial hierarchy collapses. This follows for example from the results of Dell and van Melkebeek [8] regarding lower bounds for the compressibility of the satisfiability problem, since there is an immediate reduction from SAT to ILPF(n). Similarly, it follows also from earlier results of Dom et al. [9] who showed that HITTING SET parameterized by the universe size admits no polynomial kernelization under the same assumption.

Note that both SAT and HITTING SET admit polynomial kernels with respect to number of variables respectively universe size when clauses respectively sets are restricted to constant arity (i.e., to bounded size). Concretely, the kernels have size

[2] The interested reader is referred to the online documentation and manual of ILOG CPLEX 12.4 at http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r4/index.jsp (see "presolve", "preprocessing").

[3] If the instance is larger than $\mathcal{O}(n^{cn})$, then Kannan's algorithm runs in polynomial time and we may simply return the answer or a trivial *yes*- or *no*-instance. Otherwise, the claimed bound trivially holds.