Typical Properties of Winners and Losers in Discrete Optimization

Rene Beier[*]
Max-Planck-Institut für Informatik
Saarbrücken

Berthold Vöcking[†]
Department of Computer Science
RWTH Aachen

Abstract

We present a probabilistic analysis for a large class of combinatorial optimization problems containing, e.g., all *binary optimization problems* defined by linear constraints and a linear objective function over $\{0,1\}^n$. Our analysis is based on a semi-random input model that takes care for the combinatorial structure of the underlying optimiziation problem by parameterizing which input numbers are of stochastic and which are of adversarial nature. This input model covers various probability distributions for the choice of the stochastic numbers and includes *smoothed analysis* with Gaussian and other kinds of perturbation models as a special case. In fact, we can exactly characterize the smoothed complexity of binary optimization problems in terms of their worst-case complexity: A binary optimization problem has *polynomial smoothed complexity* if and only if it has pseudopolynomial complexity.

Our analysis is centered around structural properties of binary optimization problems, called *winner*, *loser*, and *feasibility gap*. We show, if the coefficients of the objective function are stochastic, then there usually exist a polynomial size $n^{-\Omega(1)}$ gap between the best and the second best solution. Furthermore, we show if the coefficients of the constraints are stochastic then the slack of the optimal solution with respect to this constraint has usually polynomial size $n^{-\Omega(1)}$ as well. We exploit the properties of these gaps in form of an adaptive rounding scheme increasing the accuracy of calculation until the optimal solution is found. The strength of our techniques is illustrated by applications to various NP-hard optimization problems from mathematical programming, network design, and scheduling for which we obtain the first algorithms with polynomial smoothed/average-case complexity.

1 Introduction

Many combinatorial optimization problems have an objective function or constraints specified in terms of real numbers representing natural quantities like time, weight, distance, or utility. This includes some well-studied optimization problems like, e.g., traveling salesperson, shortest path, minimum spanning tree as well as various scheduling and packing problems. When analyzing the complexity of algorithms for such problems, we usually assume that these numbers are integers or rational numbers with a finite length representation. The hope is that it suffices to measure and compute with some bounded precision in order to identify an optimal or close to optimal solution. In fact, if real numbers occur only in the objective function and if this objective function is well-behaved (e.g., a linear function) then calculating with reasonable approximations of the input numbers yields a feasible solution whose objective value is at least close to the optimal objective value. More problematically, however, if the constraints are defined by real numbers, then calculating with rounded input numbers might miss all interesting solutions or might even produce infeasible solutions.

[*]Email: rbeier@mpi-sb.mpg.de. Supported by the DFG studies program GRK 623.

[†]Email: voecking@cs.rwth-aachen.de. Supported in part by DFG grant Vo889/1-2.

How can one solve optimization problems (efficiently) on a computer when not even the input numbers can be specified exactly? – In practice, optimization problems in which real numbers occur in the input are solved by simply rounding the real numbers more or less carefully. Fortunately, this approach seems to yield reasonable results. We seek for a theoretically founded explanation why this rounding approach usually works. Studying this issue under worst case assumptions does not make very much sense as, in the worst case, the smallest inaccuracy might lead to an infeasible or utterly sub-optimal solution. This question needs to be studied in a stochastic model. In the following probabilistic analysis, we will show that, under some reasonable and quite general stochastic assumptions, one can usually round real-valued input numbers after only a logarithmic number of bits without changing the optimal solution. In fact, our probabilistic analysis goes far beyond the point of explaining phenomena occurring in practice. We are able to provide algorithms with polynomial average-case complexity (more precisely, polynomial smoothed complexity) for a quite general class of discrete optimization problems. Our analysis covers various well-studied NP-hard discrete optimization problems from mathematical programming, network design, and scheduling like, e.g., multi-dimensional knapsack, constrained spanning tree, or scheduling to minimize the weighted number of tardy jobs.

1.1 A semi-random input model for discrete optimization problems

We consider optimization problems defined over a vector of n binary variables $x = (x_1, \ldots, x_n)$. The set of feasible solutions is described by the intersection of a ground set of solutions $\mathcal{S} \subseteq \{0,1\}^n$ and solutions that satisfy linear constraints of the form $w^T x \leq t$ or $w^T x \geq t$. The ground set $\mathcal{S}$ of solutions can be specified arbitrarily, for instance by linear constraints but also by non-convex conditions. While the part that specifies $\mathcal{S}$ is adversarial, we assume that the coefficients in the additional linear constraints are random or randomly perturbed real numbers. The reason for distinguishing stochastic and adversarial part of the input is that we do not want that the randomization destroys the combinatorial structure of the underlying optimization problem. A similar choice between stochastic and adversarial applies to the objective function. If it is chosen to be adversarial then our model covers arbitrary functions $f : \mathcal{S} \to \mathbb{R}$. Our probabilistic analysis, however, can also handle stochastic objective functions that are linear, that is, the objective is of the form *minimize* (or *maximize*) $c^T x$, where c is a vector of random and randomly perturbed real valued coefficients $c_1, \ldots, c_n$. In the following, we use the phrase *stochastic expression* as a generic term for the linear expressions $c^T x$ and $w^T x$ occurring in the objective function and the constraints, respectively. The number of stochastic expressions is denoted by $k \geq 1$ and the number of stochastic constraints by $k' \in \{k-1, k\}$, depending on whether or not the objective function is stochastic. For $k' \geq 1$ let $\mathcal{B}_j$ denote the set of solutions that satisfy the jth constraint, for all $j \in [k']$. The set of feasible solutions for a given problem instance is then $\mathcal{S} \cap \mathcal{B}_1 \cap \ldots \cap \mathcal{B}_{k'}$ The coefficients in the stochastic expressions are specified by independent continuous probability distributions with domain $\mathbb{R}$. Different coefficients might be drawn according to different distributions. The only restriction on these distributions is that their density function is piecewise continuous and bounded. Assuming bounded densities is necessary as otherwise worst-case instances could be approximated arbitrarily well by specifying distributions with very high density. For a given distribution, the supremum of its density function is called its *density parameter*. We will see that the maximum density parameter over the distributions of the different coefficients plays an important role in our analysis. This parameter is denoted by ϕ. Intuitively, ϕ can be seen as a measure specifying the concentration of random instances around the worst case. A worst-case instance can be interpreted as a stochastic instance in which the probability measure for each stochastic number is mapped to a single point. Thus, the larger ϕ, the closer we are to a worst-case analysis.

In our probabilistic analysis we assume that the objective function defines a unique ranking among all solutions in $\{0,1\}^n$ according to the objective function. Observe, if the objective function is stochastic then the coefficients are continuous random variables. Hence, the probability that there exist solutions with same objective value is 0. In other words, a unique ranking is given with probability 1. Recall that the objective function does not have to be linear if it is adversarial, but if it is linear, i.e., of the form $c^T x$, $c \in \mathbb{Q}^n$, then a unique ranking can always be enforced by encoding the lexicographical order among the solutions into the less significant bits of the objective function without changing the computational complexity of the underlying optimization problem by more than a polynomial factor. In fact, most of the algorithmic problems that we will study have algorithms that implicitly realize a unique ranking. In this case, one does not even need an explicit encoding. Given a unique ranking, we aim at finding the *winner*, i.e., the highest ranked solution in $\mathcal{S} \cap \mathcal{B}_1 \cap \ldots \cap \mathcal{B}_{k'}$. In the following, optimization problems satisfying all the conditions above are called *binary optimization problems with stochastic expressions* or, for short, *binary optimization problems*.

Smoothed Analysis. The framework of smoothed analysis was introduced by Spielman and Teng in [27]. They assume that first an adversary specifies all coefficients w in the constraint matrix such that the norm of each row vector is at most 1. Then these adversarial numbers are slightly perturbed by adding a random number drawn according to a Gaussian distribution with mean 0 and a specified standard deviation $\sigma > 0$. Spielman and Teng prove a running time for the Simplex algorithm under the shadow vertex pivot rule that is polynomial in the number of variables and constraints as well as in $\frac{1}{\sigma}$. Similar results have been obtained for other variants of the Simplex algorithm as well as for a few other problems [2, 4, 5, 7, 28]. Our probabilistic analysis is not restricted to the model of smoothed analysis, but we use this nice framework to illustrate our results.

We generalize smoothed analysis as follows. At first, we do not necessarily perturb all input numbers but only the coefficients in the stochastic expressions. Initially, an adversary chooses all input numbers. The adversarial choice for the coefficients that shall be stochastic is restricted to real numbers from $[0,1]$ or $[-1,1]$, depending on whether the domain should be non-negative or also include negative numbers. Then a random perturbation slightly changes the coefficients in the stochastic constraints by adding an independent random number to each of them. These random numbers are drawn according to a specified family of probability distributions satisfying the following conditions. Let $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$ be any piecewise continuous density function such that $\sup_s(f(s)) = 1$ and $\int |s| f(s) ds$ is finite, that is, the random variable described by f has a finite expected absolute value. Function f is called the *perturbation model*. For $\phi \geq 1$, we define f_ϕ by scaling f, that is, $f_\phi(s) = \phi f(s\phi)$, for every $s \in \mathbb{R}$. This way, the density parameter of f_ϕ is ϕ. We obtain ϕ-*perturbations* according to the perturbation model f by adding an independent random variable with density function f_ϕ to each stochastic input number. For example, one obtains the Gaussian perturbation model from [27] by choosing f to be the Gaussian density with standard deviation $(2\pi)^{-1/2}$. A non-negative domain can be obtained, e.g., by choosing f to be the density of the uniform distribution over $[0,1]$. In [27] the running time is described in terms of the standard deviation σ. In contrast, we describe the running time in terms of the density parameter ϕ. For the Gaussian and the uniform distribution these two parameters are closely related; in both cases, ϕ is proportional to $\frac{1}{\sigma}$.

Let us illustrate our semi-random input model by an example. In the minimum spanning tree problem one seeks for a spanning tree in a given graph that has minimum weight. In the binary program formulation of this problem there is a variable x_e for each edge $e \in E$. Thus, n corresponds to the number of edges. A 0/1 solution x is feasible if the edges in the set $\{e \in E \,|\, x_e = 1\}$ form a spanning tree. Let $\mathcal{S}$ denote the set of all solutions satisfying this condition. The combinatorial structure described by $\mathcal{S}$ should not be touched by our randomization. It makes sense, however, to assume that the objective function is stochastic as its coefficients describe measured quantities. So we may assume that these coefficients are perturbed with uniform ϕ-perturbations, that is, each of these coefficients corresponds to the sum of an adversarial number from $[0,1]$ and an independent random number drawn uniformly from $[0,\phi^{-1}]$. In the constrained minimum spanning tree problem (see, e.g., [11]), edges do not only have weights but additionally each edge has a cost c_e. Now one seeks for the minimum weight spanning tree satisfying an additionally specified linear cost constraint $\sum_e c_e x \leq T$. This additional constraint corresponds to a subset $\mathcal{B}_1 \subseteq \{0,1\}^{|E|}$ so that now $\mathcal{B}_1 \cap \mathcal{S}$ is the set of feasible solutions. Due to the additional constraint, the problem becomes NP-hard. We will see, however, that there is an algorithm with "polynomial smoothed complexity" if either the objective function or the additional latency constraint is stochastic.

1.2 How accurately do we need to calculate?

More precisely, we ask how many bits of each stochastic input number do we need to reveal in order to determine the winner? – We say that the winner is *determined* after revealing some number of the bits, when there is only one possible candidate for the winner, regardless of the outcomes of the unrevealed bits.

Theorem 1 *Consider any instance of a binary optimization problem Π. Let $n \geq 1$ denote the number of binary variables and $k \geq 1$ the number of stochastic expressions.*

a) Suppose the expected absolute value $\mathbf{E}[|w|]$ of every stochastic coefficient w is bounded from above by $\mu > 0$. Then the number of bits in front of the floating point of any stochastic coefficient w is bounded by $O(\log(1+\mu nk))$, **whp**[1].

b) Let $\phi > 0$ denote the maximum density parameter, that is, all density functions are upper-bounded by ϕ. Then the winner is uniquely determined when revealing $O(\log(1+\phi nk))$ bits after the binary point of each stochastic coefficient, **whp**.

One can always decrease ϕ or μ without changing the problem by scaling the appropriate stochastic expression (and the respective bound, where applicable) by some factor $\gamma > 0$. As the corresponding distribution has mean $\gamma\mu$ and density parameter ϕ/γ, this kind of scaling does not change the overall number of bits that need to be revealed as $\log(\mu nk) + \log(\phi nk) = \log(\gamma\mu nk) + \log(\frac{1}{\gamma}\phi nk)$. The scaling only transfers significant bits from positions before the binary points to a position after the binary point. One can eliminate one parameter by normalizing the distributions such that one parameter becomes 1. In case of a smoothed analysis, the right way to scale the input numbers is already build into the model. According to our definitions, the density function f specifying the perturbation model has to have a finite expected absolute value. For any fixed model of perturbation, $\int |s| f(s) ds = O(1)$. In particular, the expected absolute value of the density function f_ϕ is $O(\phi^{-1})$. Taking into account that the domain of the initial adversarial choices for the stochastic coefficients is $[-1,1]$ or $[0,1]$, we observe that ϕ-perturbations yield coefficients with an expected absolute value of at most $\mu = O(1+\frac{1}{\phi})$. In order to simplify the notation, our model of smoothed analysis is restricted to density parameters $\phi \geq 1$. This leads to the following result on the overall number of bits that need to be revealed per stochastic input number.

Corollary 2 *For any fixed perturbation model f, the winner is uniquely determined when revealing $O(\log(\phi nk))$ bits of each stochastic coefficient,* **whp**.

Let us explain the concepts and ideas behind the analysis for Theorem 1. Part a) of the theorem follows simply by applying the Markov inequality to the expected absolute values of the individual coefficients. The interesting part of the theorem is stated in b). In order to identify the winner one needs to *isolate* the winner from other feasible solutions having a worse objective value. Furthermore, one needs to *separate* the winner from those infeasible solutions that have a better objective

[1] with high probability, with probability $1-(nk)^{-\alpha}$, for every fixed $\alpha > 0$

value than the winner. Our analysis is based on a *generalized Isolating Lemma* – i.e., a generalization of the well-known Isolating Lemma by Mulmuley, Vazirani and Vazirani [20] – and a novel *Separating Lemma*.

The Isolating Lemma was originally presented in an article about RNC algorithms for perfect matchings [20]. It is known, however, that the lemma does not only apply to the matching problem but to general binary optimization problems with a linear objective function. The lemma states that the optimal solution of a binary optimization problem is unique with probability at least $\frac{1}{2}$ when choosing the coefficients of the objective function independently, uniformly at random from the set $\{1,2,\ldots,2n\}$. This is a very surprising and counterintuitive result as there might be an exponential number of feasible solutions whose objective values fall all into a polynomially large set, namely the set $\{1,2,\ldots,2n^2\}$, so that one can expect that an exponential number of solutions are mapped to the same objective value. The reason why the winner nevertheless is isolated is that the objective values of different solutions are not independent but the solutions represent subsets over a ground set of only n random numbers. We adapt the Isolating Lemma towards our continuous setting and generalize it towards piecewise continuous probability distributions as described in Section 1.1. In particular, different coefficients may follow different continuous probability distributions. Suppose only the objective function is stochastic, and the feasible region is fixed arbitrarily. Let ϕ denote the maximum density parameter over all coefficients in the objective functions. Define the *winner gap* to be the difference between the objective value of the winner and the second-best feasible solution, provided there are at least two feasible solutions. The generalized Isolating Lemma states that the winner gap is a continuous random variable whose density function is bounded from above by $2\phi n$, and this bound is tight. From this result one can immediately derive the following lower bound on the size of the winner gap. For every $\varepsilon \in [0,1]$, the winner gap is lower-bounded by $\frac{\varepsilon}{2\phi n}$ with probability at least $1-\varepsilon$. As a consequence, it suffices to reveal only $O(\log(\phi n))$ bits of each coefficient of the objective function in order to identify the winner, **whp**.

We accompany the Isolating Lemma with a novel *Separating Lemma*, enabling us to separate the winner from infeasible solutions with better objective value than the winner. For the time being, consider any binary optimization problem in which a single constraint is stochastic. The difficulty in checking the feasibility with respect to this constraint is that it might be likely that there are many solutions that are exponentially close to the constraint hyperplane. Nevertheless, we will see that the optimal solution can be identified by inspecting only a logarithmic number of bits per input number, **whp**. The reason is that we do not need to check the feasibility of all solutions but only of some particular solutions. The *losers* are those solutions that have a rank higher than the winner but they are infeasible because of the considered constraint. The *loser gap* is defined to be the minimal amount by which a loser (except for the solution 0^n) exceeds the constraint threshold. The Separating Lemma shows that the supremum of the density function of the loser gap is at most ϕn^2. Hence, for every $\varepsilon > 0$, the loser gap is at least $\frac{\varepsilon}{\phi n^2}$ with probability at least $1-\varepsilon$. Let us try to give some intuition about this result. If there are only a few losers then one can imagine that neither of them comes very close to a random or randomly perturbed hyperplane. However, there might be an exponential number of losers. In this case, however, the winner has a relatively low rank as there is an exponential number of solutions better than the winner; but this is very unlikely if the constraint hyperplane is likely to come very close to the good solutions which correspond to the losers. Seeing it the other way around, if there are many losers then the hyperplane is likely to be relatively far away from the losers, which might intuitively explain the phenomenon described by the Separating Lemma. Besides the loser gap, we study the so-called *feasibility gap* corresponding to the slack of the optimal solution with respect to the stochastic constraint. Essentially, we prove that the density functions of loser and feasibility gaps have the same maximum supremum so that the density of the feasibility gap is lower-bounded by $\frac{\varepsilon}{\phi n^2}$ as well. In fact, our analysis for loser and feasibility gaps is heavily based on symmetry properties between them.

Let us remark that, when analyzing the winner gap, it is assumed a random objective function and a fixed feasible region. In contrast, when analyzing loser and feasibility gaps, it is assumed a random constraint or a set of random constraints instead of a random objective function. In other words, the random expressions defining the objective function and the constraints are assumed to be stochastically independent. In fact, if the feasible region and the objective function are correlated, then winner, loser, and feasibility can not be lower-bounded by a polynomial. The optimization variant of the subset-sum problem (i.e., knapsack with profits equal to weights) is a simple counterexample. Lueker [17] proved that random instances of the subset-sum problem have usually exponentially small gaps.

2 Analysis of the Gap Properties

In this section we will formally define winner, loser, and feasibility gaps and prove upper bounds on the density functions of these random variables. Before going into the details of the analysis, the term ”upper bound on the density” needs some clarification as the density of a continuous variable is not uniquely defined. A continuous random variable X is defined by its *distribution* $F_X(t) = \mathbf{Pr}[X \leq t]$. In general, the *density* f_X is any non-negative function satisfying $F_X(t) = \int_{-\infty}^{t} f_X(s)ds$. Observe that the integrand is not uniquely determined. It might be redefined on any set of points of measure 0 without affecting the integral. We say that a continuous random variable X is *well-behaved* if its distribution function F_X is piecewise differentiable. In this case, X admits a piecewise continuous density function f_X which at all of its continuous points corresponds to the derivative of F_X. As usual, we ignore the trifling indeterminacy in the definition of f_X and refer to f_X as *the* density of X. In particular, the *supremum of the density of* X refers solely to the supremum over the points at which f_X is continuous, and we say that the density is *bounded* if there exists $b \in \mathbb{R}$ such that $f_X(s) \leq b$, for every point $s \in \mathbb{R}$ at which f_X is continuous. Throughout the analysis $[n]$ denotes $\{1,\ldots,n\}$. Before we turn to the winner, loser, and feasibility gaps let us prove the following lemma which we will frequently apply in our analysis.

Lemma 3 *Let $X_1,\ldots,X_n$ and X denote well-behaved continuous random variables. Suppose X always takes a value equal to one of the values of the variables $X_1,\ldots,X_n$. Then for all $t \in \mathbb{R}$, $f_X(t) \leq \sum_{i\in[n]} f_{X_i}(t)$.*

Proof. Let $X_1,\ldots,X_n$ and X denote well-behaved continuous random variables. We have to prove, $f_X(t) \leq \sum_{i\in[n]} f_{X_i}(t)$, for all $t \in \mathbb{R}$. For every $T \subseteq \mathbb{R}$,

$$\mathbf{Pr}[X \in T] \;\leq\; \mathbf{Pr}[\exists i \in [n] : X_i \in T] \;\leq\; \sum_{i\in[n]} \mathbf{Pr}[X_i \in T] \quad .$$

Hence, for every continuous point $t \in \mathbb{R}$,

$$f_X(t) \;=\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}[X \in [t,t+\varepsilon]]}{\varepsilon} \;\leq\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \sum_{i\in[n]} \frac{\mathbf{Pr}[X_i \in [t,t+\varepsilon]]}{\varepsilon} \;=\; \sum_{i\in[n]} \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}[X_i \in [t,t+\varepsilon]]}{\varepsilon} \;=\; \sum_{i\in[n]} f_{X_i}(t) \quad .$$

This concludes the proof of Lemma 3. □

2.1 The Winner Gap

We consider an instance of a discrete optimization problem whose solutions are described by n binary variables $x_1,\ldots,x_n$. The set of feasible solutions (ground set intersected with the constraints) is now denoted by $\mathcal{S} \subseteq \{0,1\}^n$. Fix some arbitrary set $\mathcal{S}$ with at least two solutions. The objective function is denoted by $c^T x$. The numbers $c_i \in \mathbb{R}$, $i = 1,\ldots,n$, are assumed to be stochastic, that is, they are treated as independent random variables following possibly different, well-behaved continuous probability distributions with bounded density. Without loss of generality, we consider a maximization problem. Let $x^* = \text{argmax}\{c^T x \,|\, x \in S\}$ denote the winner and $x^{**} = \text{argmax}\{c^T x \,|\, x \in S \setminus \{x^*\}\}$ the second best solution. The *winner gap* Δ is defined to be the difference between the objective values of a best and a second best solution, that is,

$$\Delta \;=\; c^T x^* - c^T x^{**} \;\; .$$

The random variable Δ is well-behaved, i.e., Δ admits a piecewise continuous density function. This can be seen as follows. The probability space of Δ is $(c_1 \times \ldots \times c_n) \subseteq \mathbb{R}^n$. Each pair of solutions defines a hyperplane in $\mathbb{R}^n$, consisting of all points where the two solutions have the same objective function value. These hyperplanes partition $\mathbb{R}^n$ into a finite number of polyhedral cells. Fix any cell $C \subseteq \mathbb{R}^n$. In this cell, x^* and x^{**} are uniquely determined. In particular, $(\Delta|C) = c^T x^* - c^T x^{**}$. Thus, the random variable $\Delta|C$ is a linear functional of the well-behaved continuous variables $c_1, \ldots, c_n$. Thus the density of $\Delta|C$ corresponds to the convolution of piecewise-continuous variables, and hence, it is piecewise continuous, too. Consequently, $f_\Delta = \sum_C \mathbf{Pr}[C]\, f_{\Delta|C}$ is piecewise continuous as well, so that Δ is well-behaved. The same kind of argument applies to other gap variables that we will define in the following.

Lemma 4 (Generalized Isolating Lemma) *Let ϕ_i denote the density parameter of c_i, $1 \le i \le n$, and $\phi = \max_i \phi_i$. For every choice of the feasible region $\mathcal{S}$ and every choice of the probability distributions of $c_1, \ldots, c_n$, the density function of Δ is bounded from above by $2\sum_{i\in[n]} \phi_i \le 2\phi n$.*

Proof. At first we observe, if there is a variable x_i that takes the same value in all feasible solutions, then this variable does not affect the winner gap and it can be ignored. Thus, without loss of generality, for every $i \in [n]$, there are at least two feasible solutions whose vectors differ in the *i*-th bit, i.e., with respect to the *i*-th variable. Under this assumption, we can define the winner gap with respect to bit position $i \in [n]$ by

$$\Delta_i \;\;=\;\; c^T x^* - c^T y \;\;|\;\; x^* \;=\; \text{argmax}\{c^T x \,|\, x \in \mathcal{S}\},\; y \;=\; \text{argmax}\{c^T x \,|\, x \in \mathcal{S}, x_i \neq x_i^*\} \;\; . \qquad (1)$$

In words, Δ_i is the difference between the objective value of the winner x^* and the value of a solution y that is best among those solutions that differ in the *i*-th bit from x^*, i.e., the best solution in $\{x \in \mathcal{S} \,|\, x_i \neq x_i^*\}$.

Clearly, the best solution, $x^* = (x_1^*, \ldots, x_n^*)$, and the second best solution, $x^{**} = (x_1^{**}, \ldots, x_n^{**})$, differ in at least one bit, that is, there exists $i \in [n]$ such that $x_i^* \neq x_i^{**}$. If the best and the second best solution differ in the *i*-th bit then $\Delta = \Delta_i$. Thus, Δ is guaranteed to take a value also taken by at least one of the variables $\Delta_1, \ldots, \Delta_n$. Observe that the random variables $\Delta_1, \ldots, \Delta_n$ are well-behaved continuous, but there might be various kinds of dependencies among these variables. In the following, we will prove an upper bound of $2\phi_i$ on the density function for the random variable Δ_i, for every $i \in [n]$. Combining this bound with Lemma 3 immediately yields an upper bound of $2\sum_{i\in[n]} \phi_i$ for the density function of Δ, and the lemma is shown.

Let us fix an index $i \in [n]$. It only remains to be shown that the density of Δ_i is bounded from above by $2\phi_i$. We partition $\mathcal{S}$, the set of feasible solutions, into two disjoint subsets $\mathcal{S}_0 = \{x \in \mathcal{S} \,|\, x_i = 0\}$ and $\mathcal{S}_1 = \{x \in \mathcal{S} \,|\, x_i = 1\}$. Now suppose all random variables $c_k, k \neq i$ are fixed arbitrarily. Obviously, under this assumption, the winner among the solutions in $\mathcal{S}_0$ and its objective value are fixed as the objective values of the solutions in $\mathcal{S}_0$ do not depend on c_i. (In fact, there can be many solutions with maximum objective value.) Although the objective values of the solutions in $\mathcal{S}_1$ are not fixed, the winner of $\mathcal{S}_1$ is determined as well because the unknown outcome of the random variable c_i does not affect the order among the solutions in $\mathcal{S}_1$. For $j \in \{0,1\}$, let $x^{(j)}$ denote a winner among the solutions in $\mathcal{S}_j$. We observe $\Delta_i = |c^T x^{(1)} - c^T x^{(0)}|$ because the solutions x^* and y as defined in Equation (1) cannot be contained in the same set $\mathcal{S}_j$, $j \in \{0,1\}$. Hence, Δ_i takes either the value of $c^T x^{(1)} - c^T x^{(0)}$ or the value of $c^T x^{(0)} - c^T x^{(1)}$. Observe that the random variable

c_i appears as a simple additive term in both of these expressions, and the density of c_i is at most ϕ_i. Therefore, both expressions describe random variables with density at most ϕ_i as well. (Observe that this holds, regardless of whether we assume that the other variables are fixed or random numbers.) Consequently, Lemma 3 yields that the density of Δ_i is at most $2\phi_i$. This completes the proof of the Generalized Isolating Lemma. □

Next we show that the given bound in the Generalized Isolating Lemma is tight.

Lemma 5 *Let ϕ_i denote the density parameter of c_i, $1 \le i \le n$. For every choice of $\phi_1, \ldots, \phi_n$, there is a way to define $\mathcal{S}$ and the distributions of $c_1, \ldots, c_n$, such that the maximum of the density function of Δ is at least $2\sum_{i\in[n]} \phi_i$.*

Proof. For every choice of n and $\phi_1, \ldots, \phi_n$, we have to present an example of an optimization problem with a stochastic optimization function $c^T x$, $c \in \mathbb{R}^n$, and a feasible region $\mathcal{S}$ such that ϕ_i is the supremum of the density function describing coefficient c_i, and the supremum of the density function of Δ is at least $2\sum_{i\in[n]} \phi_i$. Let us simply define $\mathcal{S} = \{0,1\}^n$. Then the optimal solution x^* satisfies $x^*_i = 1 \Leftrightarrow c_i > 0$. Let $k = \operatorname{argmin}_i(|c_i|)$. The second best solution x^{**} satisfies $x^{**}_i = x^*_i$, for all $i \ne k$, and $x^{**}_k = 1 - x^*_k$. Thus, $\Delta = \min_i(|c_i|)$. Assume the density functions of $c_1, \ldots, c_n$ are continuous around 0. Then

$$f_\Delta(0) \;=\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[\Delta \in [0,\varepsilon]\right]}{\varepsilon} \;=\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[\exists i \in [n] : |c_i| \in [0,\varepsilon]\right]}{\varepsilon} \;=\; \sum_{i\in[n]} \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[|c_i| \in [0,\varepsilon]\right]}{\varepsilon}$$

The last equality is due to the fact that for any $S \subset \mathcal{S}$ with $|S| > 1$,

$$\lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[\bigwedge_{i\in S} |c_i| \in [0,\varepsilon]\right]}{\varepsilon} = 0 \ .$$

Hence,

$$\lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[|c_i| \in [0,\varepsilon]\right]}{\varepsilon} \;=\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[c_i \in [0,\varepsilon]\right]}{\varepsilon} + \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[c_i \in [-\varepsilon,0]\right]}{\varepsilon} \;=\; 2 f_{c_i}(0) \ .$$

Finally, if we assume that the density functions of the coefficients take their maximum at 0, then $\phi_i = f_{c_i}(0)$ so that the maximum density is (at least) $2 f_\Delta(0) = 2\sum_{i\in[n]} \phi_i$, which completes the proof. □

2.2 Loser and Feasibility Gaps for a Single Constraint

We consider an instance of an optimization problem over n binary variables. The objective function can be fixed arbitrarily; we rank all solutions (feasible and infeasible) according to their objective value in non-increasing order. Solutions with the same objective values are ranked in an arbitrary but fixed fashion. The feasible region is described by a subset $\mathcal{S} \subseteq \{0,1\}^n$ intersected with the half-space $\mathcal{B}$ described by an additional linear constraint. Without loss of generality, the constraint is of the form $w^T x \le t$. The set $\mathcal{S}$ and the threshold t are assumed to be fixed. The coefficients $w_1, \ldots, w_n$ correspond to independent random variables following possibly different, well-behaved continuous probability distributions with bounded density. The *winner*, denoted by x^*, is the solution with highest rank in $\mathcal{S} \cap \mathcal{B}$. The *feasibility gap* is defined by

$$\Gamma \;=\; \begin{cases} t - w^T x^* & \text{if } \mathcal{S} \cap \mathcal{B} \ne \emptyset, \text{ and} \\ \perp & \text{otherwise.} \end{cases}$$

In words, Γ corresponds to the slack of the winner with respect to the constraint $w^T x \le t$. Observe that x^* might be undefined as there is no feasible solution. In this case, the random variable Γ takes

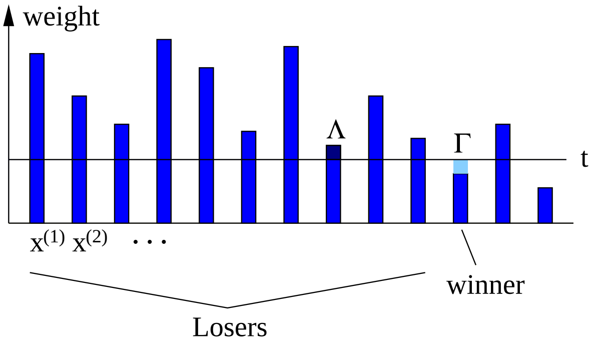

Figure 1: Loser and Feasibility gap: Solutions are listed according to the ranking. The height of the bars correspond to the weight $w^T x$. The winner is the first solution whose weight obeys the threshold t. The feasibility gap is the difference between threshold t and the weight of the winner. All solutions left of the winner are losers. The loser gap is the smallest amount by which any loser violates the threshold t.

the value $\perp$ (*undefined*). The domain of Γ is $\mathbb{R}_{\geq 0} \cup \{\perp\}$. The density function f_Γ over $\mathbb{R}_{\geq 0}$ is well-behaved continuous. The function f_Γ does not necessarily integrate to 1 but only to $1 - \mathbf{Pr}[\Gamma = \perp]$. In the following, when talking about the density of Γ, we solely refer to the function f_Γ over $\mathbb{R}_{\geq 0}$, that is, we ignore the probability of the event $\{\Gamma = \perp\}$ as it is of no relevance to us.

A solution in $\mathcal{S}$ is called a *loser* if it has a higher rank than x^*, that is, the losers are those solutions from $\mathcal{S}$ that have a better rank than the winner, but they are cut off by the constraint $w^T x \leq t$. The set of losers is denoted by $\mathcal{L}$. If there is no winner, as there is no feasible solution, then we define $\mathcal{L} = \mathcal{S}$. The *loser gap* is defined by

$$\Lambda = \begin{cases} \min\{w^T x - t \mid x \in \mathcal{L}\} & \text{if } \mathcal{L} \neq \emptyset \text{, and} \\ \perp & \text{otherwise.} \end{cases}$$

In case $\mathcal{L} \neq \emptyset$, the loser that determines Λ, i.e., $\operatorname{argmin}_{x \in \mathcal{L}} \{w^T x\}$, is called *minimal loser*. As for the feasibility gap, when talking about the density of the loser gap, we solely refer to the function f_Λ over $\mathbb{R}_{\geq 0}$ and ignore the probability of the event $\{\Lambda = \perp\}$.

Our goal is to upper-bound the densities of Γ and Λ. Observe that the solution 0^n is different from all other solutions in $\mathcal{S}$ as its feasibility does not depend on the outcome of the random coefficients $w_1, \ldots, w_n$. Suppose $0^n \in \mathcal{S}$ and 0^n has the highest rank among all solutions in $\mathcal{S}$. Then one can enforce $\Gamma = 0$ by setting $t = 0$. Similarly, one can enforce $\Lambda \to 0$ for $t \to 0$. For this reason, we need to exclude the solution 0^n from our analysis. Assuming $0^n \notin \mathcal{S}$, the following theorem shows that both the loser and the feasibility gap are likely to have polynomial size.

Lemma 6 (Separating Lemma) *Let ϕ_i denote the density parameter of w_i, for all $i \in [n]$, and define $\phi = \max_{i \in [n]} \phi_i$. Suppose $0^n \notin \mathcal{S}$. Then the densities of Γ and Λ are upper bounded by $n \sum_{i=1}^n \phi_i \leq \phi n^2$.*

Proof. We will heavily use symmetry properties between the two gaps. At first, we will prove an upper bound of ϕn on the density of the loser gap under the assumption that the ranking satisfies a certain monotonicity property. Next, we will show that the suprema of the density functions for the loser and the feasibility gap are identical for worst-case choices of the threshold t. This way, the upper bound on the density of the loser gap holds for the feasibility gap as well. Then we will show that monotonicity assumption for the feasibility gap can be dropped at the cost of an extra factor n, thereby achieving an upper bound of ϕn^2 on the density of the feasibility gap. Finally, by applying the symmetry between loser and feasibility gap again, we obtain the same result for the loser gap.

A ranking of the solutions is called *monotone* if all pairs of solutions $x, y \in \mathcal{S}$, x having a higher rank than y, satisfy that there exists $i \in [n]$ with $x_i > y_i$. When considering the binary solution vector as subsets of $[n]$, a ranking is *monotone* if each subset S is ranked higher than all its proper subsets $T \subset S$. This property is naturally satisfied for maximization problems having a linear objective function with positive coefficients, but also if all solutions in $\mathcal{S}$ have the same number of ones.

Lemma 7 *Suppose $0^n \notin \mathcal{S}$ and the ranking is monotone. Then f_Λ is bounded from above by $\sum_{i \in [n]} \phi_i$.*

Proof. Fix $t \in \mathbb{R}$ arbitrarily. As in the proof for the winner gap, we define n random variables $\Lambda_1, \ldots, \Lambda_n$ with maximum densities $\phi_1, \ldots, \phi_n$ such that at least one of them takes the value of Λ. For $i \in [n]$, define $\mathcal{S}_i = \{x \in \mathcal{S} \mid x_i = 1\}$ and $\overline{\mathcal{S}_i} = \mathcal{S} \setminus \mathcal{S}_i$. Let $\overline{x}^{(i)}$ denote the winner from $\overline{\mathcal{S}_i}$, i.e., the solution with highest rank in $\overline{\mathcal{S}_i} \cap \mathcal{B}$. Now let $\mathcal{L}_i$ denote the set of losers from $\mathcal{S}_i$ with respect to $\overline{x}^{(i)}$, that is, $\mathcal{L}_i = \{x \in \mathcal{S}_i \,|\, x \text{ has a higher rank than } \overline{x}^{(i)}\}$. If $\overline{x}^{(i)}$ does not exist then we set $\mathcal{L}_i = \mathcal{S}_i$. Now define the minimal loser of $\mathcal{L}_i$, $x^{(i)}_{\min} = \operatorname{argmin}\{w^T x \,|\, x \in \mathcal{L}_i\}$, and

$$\Lambda_i \; = \; \begin{cases} w^T x^{(i)}_{\min} - t & \text{if } \mathcal{L}_i \neq \emptyset \text{, and} \\ \perp & \text{otherwise.} \end{cases}$$

Observe that $\mathcal{L}_i$ is not necessarily a subset of $\mathcal{L}$ as $\overline{x}^{(i)}$ can have a lower rank than x^*. In fact, $x^{(i)}_{\min}$ can be feasible so that Λ_i can take negative values. The reason for this "wasteful" definition is that it yields some kind of independence that we will exploit in the following arguments.

Claim A: The density of Λ_i is at most ϕ_i.
This claim can be seen as follows. The definitions above ensure $\mathcal{L}_i \subseteq \mathcal{S}_i$ while $\overline{x}^{(i)} \in \overline{\mathcal{S}_i}$. Suppose all variables $w_j, j \neq i$ are fixed arbitrarily. We prove that the density of Λ_i is bounded by ϕ_i under this assumption, and hence the same bound holds for randomly chosen $w_j, j \neq i$ as well. The winner $\overline{x}^{(i)}$ can be determined without knowing the outcome of w_i as $\overline{x}^{(i)} \in \overline{\mathcal{S}_i}$ and for all solutions in $\overline{\mathcal{S}_i}$ the i-th entry is zero. Observe that $\mathcal{L}_i$ is fixed as soon as $\overline{x}^{(i)}$ is fixed, and so is $x^{(i)}_{\min}$, as the i-th bit of all losers in $\mathcal{L}_i$ is one. Hence, w_i is not affected by the determination of $x^{(i)}_{\min}$. As the i-th bit of $x^{(i)}_{\min}$ is set to one, the random variable Λ_i can be rewritten as $\Lambda_i = w^T x^{(i)}_{\min} - t = \kappa + w_i$, where κ denotes a fixed quantity and w_i is a random variable with density at most ϕ_i. Consequently, the density of Λ_i is bounded from above by ϕ_i.

Claim B: If $\Lambda \neq \perp$, then there exists $i \in [n]$ such that Λ takes the value of Λ_i.
To show this claim, let us first assume that x^* exists and $\mathcal{L} \neq \emptyset$. Let $x_{\min} \in \mathcal{L}$ denote the *minimal loser*, i.e., $x_{\min} = \operatorname{argmin}\{w^T x \,|\, x \in \mathcal{L}\}$. By definition, $x_{\min}$ has a higher rank than x^*. Because of the monotonicity of the ranking, there exists $i \in [n]$ such that $x^* \in \overline{\mathcal{S}_i}$ and $x_{\min} \in \mathcal{S}_i$. From $x^* \in \overline{\mathcal{S}_i}$, we conclude $x^* = \overline{x}^{(i)}$. Consequently, $x_{\min} \in \mathcal{L} \cap \mathcal{S}_i = \mathcal{L}_i$ so that $x_{\min} = x^{(i)}_{\min}$. Hence, $\Lambda = \Lambda_i$. Now suppose x^* does not exist. Then $\mathcal{L} = \mathcal{S}$ and $\mathcal{L}_i = \mathcal{S}_i$, for all $i \in [n]$. Thus, there exists $i \in [n]$ with $x_{\min} = x^{(i)}_{\min}$ because $\mathcal{S} = \bigcup_{i \in [n]} \mathcal{S}_i$ as $0^n \notin \mathcal{S}$. Finally, if $\mathcal{L} = \emptyset$ then the claim follows immediately as $\Lambda = \perp$.

Now applying Lemma 3 to the Claims A and B immediately yields the lemma. □

The following lemma shows that upper bounds on the density function of the loser gap also hold for the feasibility gap and vice versa. For a given threshold t, let $R(t) \subseteq \mathbb{R}_{\geq 0}$ denote the set of points at which the distribution functions of $\Lambda(t)$ and $\Gamma(t)$ are differentiable. As $\Lambda(t)$ and $\Gamma(t)$ are well-behaved continuous, the points in $\mathbb{R} \setminus R(t)$ have measure 0 and, hence, can be neglected.

Lemma 8 *Suppose $0^n \notin \mathcal{S}$. Then $\sup_{t \in} \; \sup_{s \in R(t)} f_{\Gamma(t)}(s) = \sup_{t \in} \; \sup_{s \in R(t)} f_{\Lambda(t)}(s)$.*

Proof. We take an alternative view on the given optimization problem. We interpret the problem as a bicriteria problem. The feasible region is then the whole set $\mathcal{S}$. The first criteria is the rank which is to be minimized (high ranks are small numbers). The second criteria is the weight, defined by the linear function $w^T x$, which is to be minimized as well. A solution $x \in \mathcal{S}$ is called *Pareto-optimal* if there is no higher ranked solution $y \in \mathcal{S}$ with smaller (or equal) weight. For simplicity assume that no two solutions have the same weight. This assumption is justified as the probability that there are two solutions with same weight is 0.

Next we show that winners and minimal losers of the original optimization problem correspond to Pareto-optimal solutions of the bicriteria problem. First, let us observe that the winner x^* with respect to any given weight threshold t is a Pareto-optimal solution for the bicriteria problem because there is no other solution with a higher rank and weight at most $t \geq w^T x^*$. Moreover, for every Pareto-optimal solution x there is also a threshold t such that x is a winner, i.e., $t = w^T x$.

The same kind of characterization holds for minimal losers as well. Recall, for a given threshold t, the minimal loser is defined to be $x_{\min} = \text{argmin}\{w^T x \,|\, x \in \mathcal{L}\}$. We claim that there is no other solution y that simultaneously achieves a higher rank and smaller weight than $x_{\min}$. This can be seen as follows. Suppose y is a solution with higher rank than $x_{\min}$. If $w^T y \leq t$ then $y \in \mathcal{B}$ and, hence, $x_{\min}$ would not be a loser. However, if $w^T y \in (t, w^T x_{\min})$ then y and $x_{\min}$ would both be losers, but y instead of $x_{\min}$ would be minimal. Furthermore, for every Pareto-optimal solution x there is also a threshold t such that x is a loser. This threshold can be obtained by setting $t \to w^T x$, $t < w^T x$.

Now let us describe winner and loser gap in terms of Pareto-optimal solutions. Let $\mathcal{P} \subseteq \mathcal{S}$ denote the set of Pareto-optimal solutions with respect to the fixed ranking and the random weight function $w^T x$. Then feasibility and loser gaps are characterized by

$$\begin{aligned} \Gamma(t) &= \min\{t - w^T x \mid x \in \mathcal{P}, w^T x \leq t\} \ , \\ \Lambda(t) &= \min\{w^T x - t \mid x \in \mathcal{P}, w^T x > t\} \ . \end{aligned}$$

For a better intuition, we can imagine that all Pareto-optimal solutions are mapped onto a horizontal line such that $x \in \mathcal{P}$ is mapped to the point $w^T x$. Then $\Gamma(t)$ is the distance from the point t on this line to the closest Pareto point *left* of t (i.e., less than or equal to t), and $\Lambda(t)$ is the distance from t to the closest Pareto point *strictly right* of t (i.e., larger than t). Now let f be a measure over $\mathbb{R}$ describing the density of the Pareto points on the line, that is, for every $t \in \mathbb{R}$,

$$f(t) = \lim_{\varepsilon \to 0} \frac{\mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in [t, t+\varepsilon]\right]}{\varepsilon}$$

Let us remark that f is not a probability density function as it does not integrate to one. It is a measure for the expected number of Pareto-optimal solutions over the line, that is, $\int_{-\infty}^{t} f(r)dr$ describes the expected number of Pareto-optimal solutions in the interval $[-\infty, t]$. In the following, we will first prove $\sup_t f_{\Lambda(t)}(0) = f(t) = \sup_t f_{\Gamma(t)}(0)$, and afterwards we will show, for every $s \in \mathbb{R}$, $\sup_t f_{\Lambda(t)}(s) \leq \sup_t f_{\Lambda(t)}(0)$ as well as $\sup_s f_{\Gamma(t)}(s) \leq \sup_t f_{\Gamma(t)}(0)$. Combining these equations proves the theorem.

We start by showing $f_{\Lambda(t)}(0) = f(t) = f_{\Gamma(t)}(0)$, for any $t \in \mathbb{R}$. The distribution function of the loser gap $\Lambda(t)$ is defined by

$$F_{\Lambda(t)}(s) = \mathbf{Pr}[\Lambda(t) \leq s] = \mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in (t, t+s]\right] = \mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in [t, t+s]\right] \ .$$

Let us assume, without loss of generality, that the density $f_{\Lambda(t)}$ is defined to be the right derivative of the distribution $F_{\Lambda(t)}$. Then

$$f_{\Lambda(t)}(0) = F'_{\Lambda(t)}(0) = \lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} \frac{F_{\Lambda(t)}(\varepsilon) - F_{\Lambda(t)}(0)}{\varepsilon} = \lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} \frac{\mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in [t, t+\varepsilon]\right]}{\varepsilon} = f(t) \ ,$$

where the third equation follows because $F_{\Lambda(t)}(0) = 0$. The feasibility gap behaves in a symmetric fashion. As it describes the distance to the closest Pareto point on the left instead of the right of t, we obtain

$$f_{\Gamma(t)}(0) \;=\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in [t, t-\varepsilon]\right]}{\varepsilon} \;=\; f(t) \;\; .$$

Therefore, $f_{\Lambda(t)}(0) = f(t) = f_{\Gamma(t)}(0)$, for every $t \in \mathbb{R}$. As a consequence, $\sup_t f_{\Lambda(t)}(0) = \sup_t f(t) = \sup_t f_{\Gamma(t)}(0)$.

It remains to generalize this result towards other parameters of the density functions than 0. First, let us consider the loser gap. Λ corresponds to the distance between a given threshold t and the *closest* Pareto point that is (strictly) right of t. Consequently, the density of Λ at a given point on the line should not decrease when the threshold is moved towards this point. Formally, for every $s \in R(t)$,

$$\begin{aligned} f_{\Lambda(t)}(s) \;&=\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{F_{\Lambda(t)}(s+\varepsilon) - F_{\Lambda(t)}(s)}{\varepsilon} \\ &=\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in [t, t+s+\varepsilon]\right] - \mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in [t, t+s]\right]}{\varepsilon} \\ &\leq\; \lim_{\substack{\varepsilon\to 0\\ \varepsilon>0}} \frac{\mathbf{Pr}\left[\exists x \in \mathcal{P} : w^T x \in [t+s, t+s+\varepsilon]\right]}{\varepsilon} \;=\; f_{\Lambda(t+s)}(0) \;\; . \end{aligned}$$

Analogously, $f_{\Gamma(t)}(s) \leq f_{\Gamma(t+s)}(0)$. Thus, $\sup_t \sup_s f_{\Lambda(t)}(s) = \sup_t f_{\Lambda(t)}(0) \sup_t f_{\Gamma(t)}(0) = \sup_t \sup_s f_{\Gamma(t)}(s)$. This completes the proof of Lemma 8. □

Combining the two lemmas, we observe that the density of the feasibility gap $\Gamma(t)$ is at most $\sum_{i\in[n]} \phi_i$, provided that the ranking is monotone and $0^n \notin \mathcal{S}$. Next we extend this result towards general rankings by breaking the original problem in subproblems. We partition $\mathcal{S}$ into the sets $\mathcal{S}^{(k)} \;=\; \{x \in \mathcal{S} \mid \sum_i x_i = k\}$, for $1 \leq k \leq n$. Observe that each of these sets contains only solutions with the same number of ones, and hence, satisfies the monotonicity condition. Let $\Gamma^{(k)}(t)$ denote the feasibility gap over the set $\mathcal{S}^{(k)}$. By Lemma 7, the density of $\Gamma^{(k)}(t)$ is at most $\sum_{i\in[n]} \phi_i$, for every $t \in \mathbb{R}$. Furthermore, $\Gamma(t)$ takes the value of one of the variables $\Gamma^{(k)}(t)$, $1 \leq k \leq n$ because the winner of one of the subproblems is the winner of the original problem. As a consequence of Lemma 3, the density of $\Gamma(t)$ is at most $n \sum_{i\in[n]} \phi_i$, for every continuous point $t \in \mathbb{R}$. Let us remark that such a kind of argument cannot directly be applied to the loser gap. By applying Lemma 8, however, the bound for the feasibility gap holds for the loser gap as well. Hence, Lemma 6 is shown. □

2.3 Loser and Feasibility Gap for Multiple Constraints

Assume there are $k \geq 2$ stochastic constraints. Without loss of generality, these constraints are of the form $w_j^T x \leq t_j$, for $j \in [k]$, and the sets of solutions satisfying these constraints are $\mathcal{B}_1, \ldots, \mathcal{B}_k$, respectively. We generalize the definition of feasibility and loser gap as follows. Given a set of solutions $\mathcal{S} \subseteq \{0,1\}^n$ and a ranking, the *winner* x^* is the highest ranked solution in $\mathcal{S} \cap \mathcal{B}_1 \cap \ldots \cap \mathcal{B}_k$. The *feasibility gap for multiple constraints* is the minimal slack of x^* over all stochastic constraints, that is, $\Gamma = \min_{j\in[k]}\{t_j - w_j^T x^*\}$, if x^* exists, and $\Delta = \perp$, otherwise.

The set of *losers* $\mathcal{L}$ consists of all solutions from $\mathcal{S}$ that have a rank higher than x^*. Observe that a loser only needs to be infeasible with respect to one of the k constraints. In particular, it is not true that the weight values of each loser are likely to be far away from the corresponding thresholds t_j, $j \in [k]$; not even if we consider only those constraints for which the respective loser is infeasible. Fortunately, however, we do not need such a property in the application of the loser gap. For every loser, one only needs a single constraint that renders the loser infeasible. Therefore, we define the *loser gap for k constraints* by

$$\Lambda \;=\; \begin{cases} \min_{x\in\mathcal{L}} \max_{j\in[k]}\{w_j^T x - t_j\} & \text{if } \mathcal{L} \neq \emptyset \;\text{ ,and} \\ \perp & \text{otherwise.} \end{cases}$$

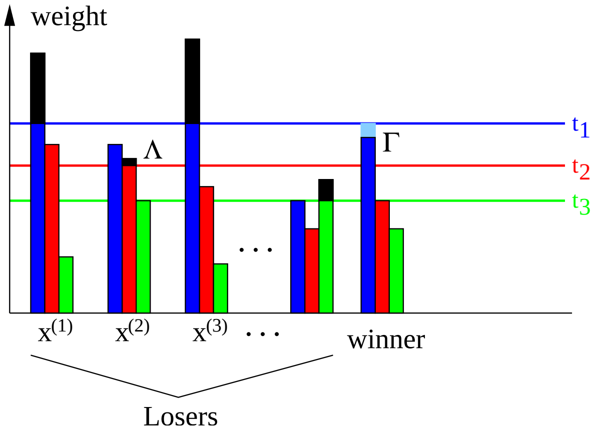

Figure 2: Loser and Feasibility gap for multiple constraints: Solutions are listed according to the ranking. Blue, red and green bars give the weight of each solution according to three different weight functions. The corresponding thresholds t_1, t_2 and t_3 are colored accordingly. The winner is the first solution obeying all thresholds. The feasibility gap is the smallest slack over the different weight functions of the winner. All solutions left of the winner are losers. The loser gap for an individual loser is the largest amount by which this loser violates any threshold (given in black). The loser gap Λ is the minimum of the individual loser gaps over all losers.

Figure 2 illustrates this definition.

Lemma 9 *Let ϕ denote the maximum density parameter of all coefficients in the stochastic constraints. Suppose $0^n \notin \mathcal{S}$. Then* $\mathbf{Pr}[\Gamma < \varepsilon] \leq \varepsilon k \phi n^2$ *and* $\mathbf{Pr}[\Lambda < \varepsilon] \leq \varepsilon k \phi n^2$*, for all* $\varepsilon \in \mathbb{R}_{\geq 0}$.

Proof. First we show the bound for the feasibility gap. Let x^* denote the winner and suppose $\Gamma \leq \varepsilon$, for some $\varepsilon \in \mathbb{R}_{\geq 0}$. Then there exists $j \in [k]$ with $t_j - w_j^T x^* \leq \varepsilon$. Thus,

$$\mathbf{Pr}[\Gamma \leq \varepsilon] \leq \sum_{j \in [k]} \mathbf{Pr}\left[t_j - w_j^T x^* \leq \varepsilon\right] \quad .$$

For each individual $j \in [k]$, we can apply the Separating Lemma assuming that the set of feasible solutions with respect to all other constraints is fixed as the coefficients in this constraint are stochastically independent from the other constraints. This way, we obtain $\mathbf{Pr}[\Gamma \leq \varepsilon] \leq k \cdot \varepsilon \phi n^2$.

Next, we turn our attention to the loser gap. Unfortunately, we cannot generalize the bound on the loser gap from one to multiple constraints in the same way as we generalized the feasibility gap as the loser gap for multiple constraints does not correspond to the minimal loser gap over the individual constraints. Instead we will make use of the result for the feasibility gap established above. Assume $\Lambda \leq \varepsilon$, for some $\varepsilon \in \mathbb{R}_{\geq 0}$. Then there exists a loser x satisfying $\forall j \in [k] : w_j^T x - t_j \leq \varepsilon$. Let x_L denote the loser with this property that is ranked highest. Consider a relaxed variant Π' of the given optimization problem Π where the thresholds of all stochastic constraints are increased by ε, i.e., we have constraints $w_j^T x \leq t_j + \varepsilon$, $j \in [k]$. Observe that x_L is feasible in the relaxed problem Π' and, by the definition of x_L, no higher ranked solution is feasible. Thus, x_L is the winner of Π'. Since $t_j < w_j^T x_L \leq t_j + \varepsilon$ for some $j \in [k]$, the feasibility gap Γ' of the relaxed problem is smaller than ε. Hence, $\Lambda \leq \varepsilon$ implies $\Gamma' \leq \varepsilon$. Finally, applying the bound $\mathbf{Pr}[\Gamma' \leq \varepsilon] \leq \varepsilon k \phi n^2$ derived in the first part of the proof yields $\mathbf{Pr}[\Lambda \leq \varepsilon] \leq \varepsilon k \phi n^2$. □

2.4 Proof of Theorem 1

First we prove part (a) of the theorem. The probability that the absolute value of any of the kn stochastic numbers $w_{i,j}$, $(i \in [k], j \in [n])$, is larger than $\mu(nk)^{1+\alpha}$, for arbitrary $\alpha > 0$, is

$$\mathbf{Pr}\left[\exists(i,j) : |w_{i,j}| > \mu(nk)^{1+\alpha}\right] \leq \sum_{i,j} \mathbf{Pr}\left[|w_{i,j}| > \mu(kn)^{1+\alpha}\right] \leq \sum_{i,j} \frac{1}{(kn)^{1+\alpha}} = (nk)^{-\alpha}.$$

Next we prove part (b) of the theorem. First, suppose that the objective function is the only stochastic expression. We reveal b bits after the binary point of each coefficient c_i $(1 \leq i \leq n)$. Then we know the value of each c_i up to a absolute error of 2^{-b}. We will deal with this lack of precise information in terms of rounding, that is, we will think of rounding down all c_i to the next multiple of 2^{-b} causing a "rounding error" of less than 2^{-b} for each number.

Lemma 10 *Let ϕ denote the maximum density parameter over the coefficients $c_1, \ldots, c_n$ in the objective function. When revealing b bits after the binary point of each coefficient then the winner is uniquely determined with probability at least $1 - 2n^2\phi/2^b$.*

Proof. Let $\lfloor c \rfloor$ be the vector that is obtained by rounding each entry c_i of vector c down to the next multiple of 2^{-b}. Consider any two solutions $x, x' \in \mathcal{S}$. We have

$$|(c^T x - c^T x') - (\lfloor c \rfloor^T x - \lfloor c \rfloor^T x')| = |(c - \lfloor c \rfloor)^T (x - x')| < n2^{-b} \quad ,$$

as $c_i - \lfloor c_i \rfloor < 2^{-b}$, for all $i \in [n]$. Hence, if the winner gap Δ (with respect to the exact coefficients $c_1, \ldots, c_n$) is at least $n2^{-b}$ then the rounding can not affect the optimality of the winner. In this case the winner is uniquely determined after revealing only b bits of each coefficient c_i. Let ϕ_Δ denote the supremum of the density of Δ. Since Δ is non-negative, $\mathbf{Pr}[\Delta < x] \leq x\phi_\Delta$, for all $x \in \mathbb{R}_{\geq 0}$. Using the generalized Isolating Lemma we obtain $\phi_\Delta \leq 2\phi n$. Setting $x = n2^{-b}$ yields $\mathbf{Pr}\left[\Delta < n2^{-b}\right] \leq \frac{2n^2\phi}{2^b}$. □

Next suppose only some of the constraints are stochastic and the objective function is adversarial. Let k' denote the number of stochastic constraints. Without loss of generality, the constraints are of the form $w_j^T x \leq t_j$, $j \in [k']$. Assume we reveal b bits after the binary point of each coefficient in the k' constraints.

Lemma 11 *Let ϕ denote the maximum density parameter over all coefficients in the stochastic constraints. When revealing b bits after the binary point of each coefficient, then the winner is uniquely determined with probability at least $1 - 2^{-b}k'n^3\phi$.*

Proof. As we round down, infeasible solutions might become feasible whereas feasible solutions stay feasible. (For constraints of the form $w_j^T x \geq t_j$ one would need to round up to the next multiple of 2^{-b}.) To ensure that the winner is uniquely determined it suffices to upper bound the maximum possible error in each constraint caused by the rounding of the coefficients. If this error is smaller than the loser gap then rounding cannot change the feasibility status of any loser, i.e., all infeasible solutions that have rank higher than the winner stay infeasible.

In order to apply the bound on the loser gap given in Lemma 9, let us first assume $0^n \notin \mathcal{S}$. The rounding error in each expression is at most $n2^{-b}$. The definition of the loser gap for multiple stochastic constraints states that for every loser x there is a constraint $j \in [k']$ such that $w_j^T x - t_j \geq \Gamma$. Therefore, if $\Gamma \geq n2^{-b}$ then every loser stays infeasible with respect to at least one constraint after rounding. Applying Lemma 9, the probability for this event is at least $1 - k'\phi n^3/2^b$.

The solution 0^n can influence the loser gap in two ways. At first, 0^n might belong to the set of losers and thus decrease the loser gap. However, rounding the coefficients w_i does not change the objective value of this solution. Thus, 0^n stays infeasible under rounding and the loser gap with respect to all other solutions in unaffected. At second, 0^n might be the winner, which would result in a different loser set when including 0^n to $\mathcal{S}$. In this case, however, 0^n has a higher rank than the previous winner so that the set of losers can only shrink. Therefore, the solution 0^n cannot decrease the loser gap and the probability that any loser becomes feasible can be estimated with the Separating Lemma as before. □

Now suppose some constraints and the objective function are stochastic. Let $k = k' + 1$ denote the number of stochastic expressions. The probability that winner and loser gap are sufficiently large, as described in the two lemmas above, is $1 - k'\phi n^3/2^b - 2\phi n^2/2^b \geq 1 - k\phi n^3/2^b$, for $n \geq 2$. This implies that the winner is uniquely determined when revealing $b = O(\log(k\phi n))$ bits, **whp**. This completes the proof of Theorem 1. □

3 Characterizing Polynomial Smoothed Complexity

Based on the gap properties, we aim at characterizing which discrete optimization problems have polynomial time algorithms under random perturbations. We formalize this as follows. Fix any binary optimization problem Π and any perturbation model f. Let $\mathcal{I}_N$ denote the set of all unperturbed instances of length N that the adversary may specify. The definition of the input length N needs some clarification as the coefficients in the stochastic expressions are assumed to be real numbers. We define that each of these numbers has a virtual length of one. (This way, we ensure $N \geq kn$.) The bits of the stochastic numbers can be accessed by asking an oracle in time $O(1)$ per bit. The bits after the binary point of each coefficient are revealed one by one from left to right. The deterministic part of the input does not contain real numbers and can be encoded in an arbitrary fashion. For an instance $I \in \mathcal{I}_N$, let $I + f_\phi$ denote the random instance that is obtained by a ϕ-perturbation of I. We say that Π has *smoothed polynomial complexity* if and only if it admits an algorithm $\mathcal{A}$ whose running time T satisfies

$$\exists \alpha, \beta > 0 : \forall \phi \geq 1 : \forall N \in \mathbb{N} : \max_{I \in \mathcal{I}_N} \mathbf{E}\left[\left(T(I + f_\phi)\right)^\alpha\right] \leq \beta \phi N \ .$$

This definition of polynomial smoothed complexity follows more or less the way how polynomial complexity is defined in average-case complexity theory, adding the requirement that the running time should be polynomially bounded not only in N but also in ϕ. It is not difficult to show that the assumption on the running time of $\mathcal{A}$ is equivalent to requiring that there exists a polynomial $P(N, \phi, \frac{1}{\varepsilon})$ such that for every $N \in \mathbb{N}, \phi \geq 1, \varepsilon \in [0,1]$, the probability that the running time of $\mathcal{A}$ exceeds $P(N, \phi, \frac{1}{\varepsilon})$ is at most ε. Observe that this does not imply that the expected running time is polynomially bounded. To enforce expected polynomial running time, the exponent α in the definition of polynomial smoothed complexity should have been placed outside instead of inside the expectation. The reason for not defining polynomial smoothed complexity based on the expected running time is that this is not a sufficiently robust notion. For example, an algorithm with expected polynomial running time on one machine model might have expected exponential running time on another machine model. In contrast, the above definition yields a notion of polynomial smoothed complexity that does not vary among classes of machines admitting polynomial time simulations among each other. Although polynomial smoothed complexity does not always imply polynomial bounds on the expected running time, we will show that several of our algorithmic results yield expected polynomial running time on a RAM.

We show that the smoothed complexity of a binary optimization problem Π can be characterized in terms of the worst-case complexity of Π. Theorem 1 shows that one usually needs to reveal only a logarithmic number of bits per real-valued input number. This suggests a connection between pseudo-polynomial worst-case running time and polynomial average-case complexity. For a binary optimization problem Π, let Π_u denote the corresponding optimization problem in which all numbers in the stochastic expression are assumed to be integers in unary representation instead of randomly chosen real-valued numbers. The following theorem holds for any fixed perturbation model f.

Theorem 12 *A binary optimization problem* Π *has polynomial smoothed complexity if and only if* $\Pi_u \in \mathsf{ZPP}$.

In other words, Π has polynomial smoothed complexity if it admits a (possibly randomized) algorithm whose (expected) running time for all instances is pseudo-polynomial in the stochastic constraints and polynomial in the remaining input. Notice that the expectation is over the randomization of the algorithm, not over the instances. This characterization immediately shows that strongly NP-hard optimization problems do not have polynomial smoothed complexity, unless ZPP=NP. This observation might not sound very surprising, as the hardness of strongly NP-hard problems does not rely on large or precisely specified input numbers. Observe, however, that the strong NP-hardness of a problem does not immediately rule out the possibility of a polynomial average-case complexity. For example, the TSP problem (on a complete graph) with edge lengths drawn uniformly at random from $[0,1]$ might have a polynomial average-case complexity. Our theorem, however, shows that it does not have a polynomial smoothed complexity, unless ZPP = NP. The more sophisticated part of the theorem is the other direction stating that every binary optimization problem admitting a pseudo-polynomial time algorithm has polynomial smoothed complexity. This result is based on the Generalized Isolating Lemma and the Separating Lemma. The idea is as follows. We design efficient verifiers checking whether a solution computed with a certain precision is actually the optimal solution of Π. The success probability of these verifiers is analyzed with the help of the gap properties. Using an adaptive rounding procedure, we increase the precision until we can certify that the computed solution is optimal. The overall running time of this meta-algorithm is polynomial if the algorithm computing solutions with bounded precision has pseudo-polynomial running time.

Proof. At first, we prove that the existence of a randomized pseudo-polynomial time algorithm for a binary optimization problem Π implies polynomial smoothed complexity for Π. We design an algorithm with polynomial smoothed complexity calling the pseudo-polynomial algorithm with higher and higher precision until the solution found is certified to be optimal. We describe verifiers, that, based on the first b bits after the binary point of each coefficient in the stochastic expressions, either certify optimality or reports FAILURE, stating that it has not sufficient information to ensure optimality. So the algorithm has access to the first b bits only, which corresponds to rounding down the numbers to the next multiple of 2^{-b}. Again we will interpret the lack of precise information as a rounding error. In the following, $\lfloor w \rfloor$ denotes the value of w rounded down to the next multiple of 2^{-b}.

Certifying optimality For a moment assume that only the objective function is stochastic. Without loss of generality, consider a maximization problem with objective function $c^T x$. At first, we compute an optimal solution x' for the problem with the rounded coefficients $\lfloor c_1 \rfloor, \ldots, \lfloor c_n \rfloor$. To check whether x' is optimal with respect to the original cost vector c, we generate another vector $\bar{c}$ of rounded coefficients. This time the rounding depends on the computed solution x'. For all i with $x'_i = 1$, we set $\bar{c_i} := \lfloor c_i \rfloor$ and for all i with $x'_i = 0$, we set $\bar{c_i} := \lceil c_i \rceil = \lfloor c_i \rfloor + 2^{-b}$. Observe that the function $\delta(x) = c^T x - \bar{c}^T x$ is maximal for $x = x'$. Next we compute an optimal solution x'' for the problem with the vector $\bar{c}$. If $x' = x''$ then x' simultaneously maximizes $\delta(x)$ and $\bar{c}^T x$. Consequently, it maximizes $\bar{c}^T x + \delta(x) = c^T x$ as well and, hence, x' corresponds to the true winner x^*. Thus, the algorithm outputs x' as a certified winner if $x' = x''$ and reports FAILURE otherwise. If the winner gap is large enough so that the winner is uniquely determined in the sense of Lemma 10, then the algorithm will always compute a certified winner. Hence, the probability that the algorithm is successful is at least $1 - 2n^2\phi/2^b$, corresponding to the bound given in Lemma 10. Observe that

$\Gamma \geq n2^{-b}$ is a sufficient but not a necessary condition to certify the optimality of the winner with b revealed bits per coefficient. So the verifier might verify optimality of the computed solution even if $\Gamma < n2^{-b}$.

Certifying feasibility Now we show how to deal with stochastic constraints. Without loss of generality, we assume that all stochastic constraints are of the form $w_j^T x \leq t_j$, $1 \leq j \leq k'$. For constraints of the form $w^T x \geq t$, we would need to round up instead of rounding down. First we compute a certified winner, denoted by x', using the rounded coefficients in the stochastic constraints. If the objective function is not stochastic, then there is no need to certify the winner. As we round down all coefficients in the stochastic constraints we ensure that feasible solutions stay feasible. However, we have to detect infeasible solutions that become feasible due to the rounding and displace the true winner. Hence, we need to check whether x' is indeed feasible with respect to the original constraints. This would be trivial if the exact values of all constraint vectors $w_1, \ldots, w_k$ were available. However, we want to check the feasibility using only the knowledge of the first b bits after the binary point of each coefficient. Assume the solution x is infeasible with respect to the j-th constraint and becomes feasible due to rounding. Then $\lfloor w_j \rfloor^T x \leq t_j < w_j^T x$ and hence $t_j - \lfloor w_j \rfloor^T x < w_j^T x - \lfloor w_j \rfloor^T x \leq n2^{-b}$, i.e. the slack of x in the j-th constraint is less than $n2^{-b}$ with respect to the rounded vector $\lfloor w_j \rfloor$. Our verifier will use this property and classifies x' as possibly infeasible if it has slack less than $n2^{-b}$ for any of the k constraints with respect to the rounded coefficients. This way, we have two failure sources. At first, there might be a loser that becomes feasible because of rounding. As seen in the proof of Lemma 11, this can happen only if the loser gap is smaller than $n2^{-b}$. At second, the true winner can be rejected since its slack is less than $n2^{-b}$. This happens only if the feasibility gap is smaller than $n2^{-b}$. Applying Lemma 9 yields that the probability that one of these events happens is at most $2k'\phi n^3/2^b$.

Adaptive rounding procedure. Consider a binary optimization problem Π with n binary variables and k stochastic expressions. Assume there exists an algorithm $\mathcal{A}$ for the special variant of Π, where the domain of the coefficients in the stochastic expressions is restricted to $\mathbb{Z}$. Furthermore, assume that the worst case running time of $\mathcal{A}$ is bounded by some polynomial in W, n, k and N, where W denotes the largest absolute value of any coefficient in the stochastic expressions. Recall that N specifies the size of the deterministic part of the input. We use the following adaptive rounding scheme. We start by revealing $b = \log(k\phi n^3)$ bits of each coefficient in the stochastic expressions. We obtain integer coefficients by scaling these numbers by 2^b. Now we use the algorithm $\mathcal{A}$ to obtain a solution and use the verifiers to test for optimality. In case of FAILURE we increment b by one and iterate until the verifiers conclude optimality of the computed solution.

To analyze the running time of $\mathcal{A}$ we need to estimate W, the largest absolute value of any integer in the stochastic expressions. It is the product of two factors. The first factor, $W_1 = 2^b$, is due to the scaling and depends on the number of revealed bits after the binary point of each coefficient. The second factor W_2 corresponds to the integer part of the largest absolute value of any coefficient. This way, $W = W_1 W_2$. We have to show that there exists a polynomial $P(N, \phi, \frac{1}{\varepsilon})$ such that for every $N \in \mathbb{N}, \phi > 0, \varepsilon \in [0,1] : \mathbf{Pr}\left[T > P(N, \phi, \frac{1}{\varepsilon})\right] \leq \varepsilon$. The running time of $\mathcal{A}$ is polynomial in W, n, k and N. As n, k and N are deterministic, it suffices to show such a polynomial bound for W. Let us first prove an upper bound on W_2. Let $r_{j,i}$ be the random variable that is added to the i-th coefficient in the j-th stochastic expression when perturbing the instance. Recall that the mean of the absolute values of the probability distribution that defines the perturbation model, is a constant, denoted E. It holds $W_2 \leq \max_{j \in [k], i \in [n]} |r_{j,i}| + 1$. Hence, for every $\alpha \geq 1$,

$$\mathbf{Pr}\left[\max_{j \in [k], i \in [n]} |r_{j,i}| > \alpha E\right] \leq \sum_{j \in [k], i \in [n]} \mathbf{Pr}\left[|r_{j,i}| > \alpha E\right] \leq \frac{nk}{\alpha} \ ,$$

where the last inequality holds by an application of the Markov Inequality. Thus, $\mathbf{Pr}[W_2 > \alpha E + 1] \leq nk/\alpha$. Setting $\varepsilon = 2nk/\alpha$, it holds $W_2 \leq \alpha E + 1 = \frac{2nkE}{\varepsilon} + 1$ with probability at least $1 - \varepsilon/2$. Next consider the term $W_1 = 2^b$. There are two reasons why the checkers can declare the computed solutions to be suboptimal or infeasible. First, the random instance happens to have a small winner gap or a small loser gap such that the winner is not uniquely determined. Using Theorem 10 and 11, the probability for this event is at most $k\phi n^3/2^b$. Secondly, the feasibility checker reports false negatives due to a small feasibility gap. This happens with probability at most $k'\phi n^3/2^b$. Allowing a failure probability of $\varepsilon/4$ for each event, we obtain $W_1 = 2^b = \frac{k\phi n^3 4}{\varepsilon}$. Therefore, with probability at least $1-\varepsilon$, none of these bad events occur and it holds $W = W_1 W_2 \leq \left(\frac{2nkE}{\varepsilon} + 1\right)\frac{k\phi n^3 4}{\varepsilon}$. As k and E are assumed to be constant and $n \leq N$, there exists a polynomial $P(N, \phi, \frac{1}{\varepsilon})$ such that for all $N \in \mathbb{N}, \phi > 0, \varepsilon \in [0,1] : \mathbf{Pr}\left[W > P(N, \phi, \frac{1}{\varepsilon})\right] \leq \varepsilon$.

From polynomial smoothed complexity to pseudo-polynomial running time Finally, we need to show that polynomial smoothed complexity of a binary optimization problem Π implies the existence of a randomized pseudo-polynomial algorithm for Π. Since we are aiming for a pseudo-polynomial time algorithm, we can assume that all numbers in the stochastic expressions are integers. Let M denote the largest absolute value of these numbers. The idea is to perturb all numbers only slightly such that the perturbation changes the value of each expression by at most $\frac{1}{2}$. To ensure that the set of feasible solutions is not changed by the perturbation, we relax all constraints by $\frac{1}{2}$, i.e., we replace $w^T x \leq t$ by $w^T x \leq t + \frac{1}{2}$ for all stochastic constraints. We then use an algorithm with polynomial smoothed complexity to compute an optimal solution x^* for the perturbed problem. By bounding the error due to the random perturbation, x^* can be shown to be optimal for the original problem as well.

Let us describe the proof idea in more detail. Our smoothed analysis framework assumes that all numbers in the stochastic expressions fall into the interval $[-1,1]$ (or $[0,1]$) before they are perturbed. To adapt our problem to this framework, we first scale all input numbers in the stochastic expressions by M^{-1}. Consequently, we have to ensure that the perturbation changes the value of an expression by at most $1/(2M)$. In particular, we will allow only perturbations that change each individual number by at most $1/(2Mn)$. We call such a perturbation *proper*. For the uniform distribution, we could simply set $\phi = 2Mn$. However, we have to deal with arbitrary families of distributions, as defined in our smoothed analysis framework, and they do not necessarily have a finite domain. The idea is to choose ϕ large enough so that a random perturbation is proper with probability at least $1/2$. Recall that the perturbation model is described by the density function f with density parameter $\phi = 1$. For other values of ϕ, we scale f appropriately. By our assumptions on f, it holds $\int |t| f_\phi(t) dt = E/\phi$ for some fixed $E \in \mathbb{R}$. Let r be a random variable following f_ϕ. Setting $\phi = 4n^2 kEM$ and applying the Markov inequality yields $\mathbf{Pr}\left[|t| > \frac{1}{2nM}\right] = \mathbf{Pr}\left[|t| > \frac{2nkE}{\phi}\right] \leq \frac{1}{2nk}$. Our perturbation draws kn of these random variables. The probability that the perturbation is proper, i.e., the probability that their absolute value is at most $\frac{1}{2nM}$, is $1/2$.

Consider any binary optimization problem Π with polynomial smoothed complexity. Polynomial smoothed complexity implies that the problem admits an algorithm $\mathcal{A}$ whose running time can be bounded polynomially in n and ϕ, with any constant probability. In particular, there exists a polynomial $P(n,\phi)$ such that the probability that the running time exceeds $P(n,\phi)$ is at most $\frac{1}{4}$. We use $\mathcal{A}$ as a subroutine in order to obtain a pseudo-polynomial algorithm. This algorithm works as follows. At first, it generates a perturbation and checks whether it is *proper*, If it is proper, then it runs $\mathcal{A}$ for at most $P(n,\phi)$ time steps. If $\mathcal{A}$ has not finished within this time bound, the algorithm returns FAILURE. Let Q be the event that the perturbation is proper. Observe that for any two events A and B it holds $\mathbf{Pr}[A \wedge B] \geq \mathbf{Pr}[A] + \mathbf{Pr}[B] - 1$. Therefore, the success probability of our algorithm is

$$\mathbf{Pr}[Q \wedge (T \leq P(n,\phi))] \;\geq\; \mathbf{Pr}[Q] - \mathbf{Pr}[T > P(n,\phi)] \;\geq\; \frac{1}{4} \;.$$

The running time of this algorithm is pseudo-polynomial because $\phi = O(Mn^2k)$. Hence, $\Pi_u \in \mathsf{ZPP}$. This completes the proof of Theorem 12. □

4 Algorithmic Applications

Let us illustrate the strength of Theorem 12 by giving some algorithmic applications to some well-known optimization problems and comparing these results with previous work on the probabilistic analysis of optimization problems. There has been substantial effort to analyze random instances of the knapsack problem, see, e.g., [5, 6, 12, 15, 16]. The knapsack problem can be seen as the simplest non-trivial binary optimization problem as its feasible region is described by only a single linear constraint. The problem belongs to the class of packing problems, that is, the constraint is of the form $w^T x \leq t$ and the coefficients are assumed to be non-negative. To our knowledge, the knapsack problem is the only NP-hard optimization problem that was previously known to have polynomial smoothed complexity [5]. The multi-dimensional knapsack problem is a natural generalization in which there are multiple packing constraints instead of only one. Dyer and Frieze [9] proved that, with constant probability, this problem can be solved in polynomial time if the number of constraints is constant and the coefficients in the constraints as well as in the objective function are chosen uniformly at random from $[0,1]$. Their result, however, does not yield polynomial average-case complexity as the dependence of the running time on the failure probability is not bounded by a polynomial. The multiple knapsack problem with a constant number of constraints admits a pseudopolynomial algorithm [14]. Hence, Theorem 12 implies polynomial smoothed and, therefore, also polynomial average-case complexity for this problem. Moreover, the pseudopolynomial algorithm also works for general 0/1 integer programming with any fixed number of constraints, i.e., when extending the domain of the coefficients to negative numbers. Therefore, this class of problems has polynomial smoothed complexity as well. This holds, even if one constraint or the objective function is assumed to be adversarial. Furthermore, the theorem shows that general 0/1 integer programming as well as the multiple knapsack problem with an unbounded number of constraints has no polynomial smoothed complexity as it is strongly NP-hard.

It is important to interpret this result in the right way. It does not prove that all problems that can be formulated as a 0/1-integer program (with a constant number of constraints) have polynomial smoothed complexity. In the formulation of these problems, variables usually appear in more than one constraint. Furthermore, the constraint matrix is usually sparse, containing many zero entries. Perturbing the constraints independently, as assumed by our smoothed analysis framework, destroys the structure of the problem. So instead of averaging over "similar" instances of the given problem, we would average over instances of some more general problem that might be easier to solve on average. We will see in Section 6 how to strengthen our analysis by allowing zero entries in the constraints which are explicitly not perturbed. This way we are able to extend our analysis to problems with sparse constraint vectors.

4.1 Scheduling Problems

The problem of scheduling to minimize the weighted number of tardy jobs is defined by n jobs each of which has a processing time p_i, a due date d_i, and a penalty c_i, which has to be paid if job i has not been finished at due date d_i. The jobs shall be scheduled on a single machine such that the sum of the penalties is minimized. In terms of n binary variables $x_1, \ldots, x_n$, the objective is to minimize $c^T x$ where $x_i = 1$ if job i is not finished in time. Observe that the problem is essentially solved once these binary variables are determined as we can assume without loss of generality that an optimal schedule executes the selected jobs in the order of non-decreasing due dates. Notice that the feasible region is completely determined by the processing times and the due dates. The objective, however, is a linear function $c^T x$. Using dynamic programming, the problem can be solved in time $O(n^2C)$, where C denotes the largest penalty for any job [22]. Hence, the problem has polynomial smoothed complexity for stochastic penalties.

4.2 Multi-Criteria Optimization Problems

If several criteria shall be optimized simultaneously then usually one of them is declared to be the objective function, and the others are formulated in form of a constraint with a given threshold. Often when a single-criteria optimization problem is polynomial, the problem becomes NP-hard when adding another criteria in form of a linear constraint. Examples for such problems are shortest path, spanning tree, or matching [19, 11, 21]. Theorem 12 enables us to prove polynomial smoothed complexity for such multicriteria problems as follows. The problems listed above have exact algorithms with pseudopolynomial running time [21, 3, 20], that is, given an integer k and an instance of these problems one can compute a solution with objective value exactly k in pseudopolynomial time. Using standard coding techniques (see, e.g., [26]) a pseudopolynomial algorithm for the exact single-criteria decision problem implies a pseudopolynomial algorithm for its multicriteria optimization variant. Combining this observation with Theorem 12 yields the following result.

Corollary 13 *Let Π be a (single objective) binary optimization problem. Suppose the exact version of Π admits an algorithm with pseudo-polynomial running time. Assume we deal with multicriteria variants of Π by choosing one criterion as the objective function and bounding the remaining criteria by appropriate thresholds. Then any multicriteria variant of Π with a constant number of criteria has polynomial smoothed complexity, provided that all criteria are stochastic.*

A similar approach was used in [21] to derive approximation schemes for multiobjective optimizations problems. The corollary implies polynomial smoothed complexity for the multicriteria variants of shortest path, spanning tree, and matching. One does not always need to assume that all criteria are of stochastic nature. For example, the bicriteria variant of the shortest path problem, i.e., the constrained shortest path problem, has an algorithm whose running time is pseudopolynomial with respect to the objective function and another algorithm that is pseudopolynomial with respect to the additional constraint. Applying Theorem 12 directly to these algorithms yields that the constrained shortest path problem has polynomial smoothed complexity even when either only the objective function or the additional constraint are stochastic.

5 Expected Polynomial Running Time

The main advantage of polynomial smoothed/average-case complexity is its robustness under different machine models. Besides, it allows a nice characterization of binary optimization problems under random inputs in terms of the problems' worst-case complexity. However, the guaranty on the running time provided by polynomial smoothed/average-case complexity is weaker than the guaranty that the expected running time is polynomial. Making additional assumptions, we can use our analysis to conclude expected polynomial running time for certain binary optimization problems.

We use again our adaptive rounding scheme, which increases the precision of the stochastic input numbers until the computed solution is certified to be optimal. We assume that the running time of the last iteration of this meta-algorithm dominates the cumulative running time of all previous iterations. In fact, it suffices if all previous iterations have cumulative running time at most a factor n^l larger than the running time of the last iteration, for some constant $l \in \mathbb{R}$. In order to obtain expected polynomial running time under random perturbations, one needs an algorithm with "pseudo-linear" instead of pseudo-polynomial running time. The only parameter in the running time that depends on the random perturbations is W, the largest absolute value of any (rounded and scaled) coefficient appearing in a stochastic expression.

We use the notation from the proof of Theorem 12. We divided W into two parts $W = W_1 W_2$, where W_1 is due to the scaling by factor $2^b = W_1$ and depends on b, the number of revealed bits after the binary point of each coefficient. The second factor W_2 corresponds to the integer part of the largest absolute value of any coefficient. In the proof of Theorem 12 we have shown the following bounds on the random variables W_1 and W_2.

$$\begin{aligned}
\mathbf{Pr}\left[W_1 > 2^b\right] &\leq 2k\phi n^3/2^b, \quad \text{for any } b \in \mathbb{N}, \text{ and} \\
\mathbf{Pr}\left[W_2 > \alpha E + 1\right] &\leq nk/\alpha\,, \qquad \text{for any } \alpha \in \mathbb{R}_{>0}.
\end{aligned}$$

Since 2^b as well as α can grow linearly with the reciprocal of the failure probability, $W \approx 2^b \alpha E$ can grow quadratically. Hence, these bounds do not allow to conclude a polynomial expected running time. Therefore, we will restrict the choice for the perturbation model by allowing only probability distributions whose tail function exhibits an exponential decay. More precisely, we assume that if X is drawn according to perturbation model f then there exists some $E \in \mathbb{R}_{\geq 0}$, such that $\mathbf{Pr}\left[|X| > \alpha E\right] \leq 2^{-\alpha}$, for every $\alpha \geq 2$. For example, the Gaussian and the exponential distribution have this property as well as all distributions with finite domain.

In the following analysis we exploit the fact that it is very unlikely that any of the coefficients in the stochastic expressions is much larger that E. Define event $\mathcal{E}$ by $(W_2 \leq n\log(nk)E + 1) \wedge (W_1 \leq 2^n)$. As $\mathbf{Pr}\left[W_2 > nlog(nk)E\right] \leq kn\,\mathbf{Pr}\left[X > n\log(nk)E\right] \leq 2^{-n}$ and $\mathbf{Pr}\left[W_1 > 2^n\right] \leq 2k\phi n^3/2^n$ it holds $\mathbf{Pr}\left[\neg\mathcal{E}\right] \leq O(k\phi n^3/2^n)$. Assume that we have a pseudolinear running time bound of $T \leq N^l W$ for some constant $l \in \mathbb{R}$. In case of $\neg\mathcal{E}$, we use a brute force enumeration of all 2^n possible $0/1$-vectors which takes time $2^n P(N)$ for some polynomial $P(N)$. Additionally, we contribute the time for the unsuccessful adaptive rounding scheme, which is at most $N^l 2^n(n\log(nk)E + 1)$, to the enumeration process. Hence, the total running time is $O(2^n P(N))$ for some other polynomial $P(N)$. Next define random variable

$$W_1' = \begin{cases} W_1 & \text{in case of } \mathcal{E}\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

The expectation of W_1' is

$$\mathbf{E}\left[W_1'\right] = \sum_{i=0}^{n} \mathbf{Pr}\left[W_1' = 2^i\right] 2^i \leq \sum_{i=0}^{n} \mathbf{Pr}\left[W_1 \geq 2^i\right] 2^i \leq \sum_{i=0}^{n} \frac{4\phi k n^3}{2^i} 2^i = O(\phi k n^4)\ .$$

Let T denote the running time of our algorithm. As $\mathbf{E}\left[W_1'\right] = \mathbf{Pr}\left[\mathcal{E}\right] \mathbf{E}\left[W_1' \mid \mathcal{E}\right]$,

$$\begin{aligned}
\mathbf{E}\left[T\right] &\leq \mathbf{Pr}\left[\mathcal{E}\right] \cdot \mathbf{E}\left[N^l W_1 W_2 \mid \mathcal{E}\right] + \mathbf{Pr}\left[\neg\mathcal{E}\right] \cdot 2^n P(N) \\
&\leq N^l(n\log(nk)E + 1) \cdot \mathbf{Pr}\left[\mathcal{E}\right] \mathbf{E}\left[W_1' \mid \mathcal{E}\right] + \mathbf{Pr}\left[\neg\mathcal{E}\right] \cdot 2^n P(N) \\
&\leq N^l(n\log(nk)E + 1)\, \mathbf{E}\left[W_1'\right] + O(k\phi n^3/2^n) \cdot 2^n P(N) \\
&= O(N^l(n\log(nk)E + 1)\, \phi k n^4 + k\phi n^3 P(N))\ .
\end{aligned}$$

Corollary 14 *Assume, the perturbation model f satisfies $\mathbf{Pr}\left[|X| > \alpha E\right] \leq 2^{-\alpha}$, for some $E \in \mathbb{R}_{\geq 0}$ and all $\alpha \geq 2$. If a binary optimization problem has pseudo-polynomial running time $O(poly(N)W)$, then this problem allows an algorithm with expected polynomial running time.*

Such pseudolinear algorithms exist on a uniform RAM, e.g., for the knapsack problem, the problem of scheduling to minimize weighted tardiness or the constrained shortest path problem. Hence, assuming the uniform RAM model, all these problems admit algorithms with expected polynomial running time under random perturbations. Observe, however, that the requirement of a pseudolinear running time is a strong restriction for binary optimization problems with $k > 1$ stochastic constraints, as standard approaches usually exhibit a running time of $\text{poly}(n)W^k$. In case of dynamic programming, for example, every additional constraint will add another dimension to the dynamic programming table.

6 Zero-Preserving Perturbations

One criticism of the smoothed analysis of the Simplex algorithm is that the additive perturbations destroy the zero-structure of an optimization problem, as it replaces zeros with small values. See also the discussion in [27]. The same criticism applies to the zero-structure in binary programs. It turns out, however, that our probabilistic analysis in Section 2 is robust enough to deal with zero-preserving perturbations. In particular, we can extend our semi-random input model introduced in Section 1.1 by allowing the coefficients in the stochastic expressions to be fixed to zero instead of being a random variable. In the model of smoothed analysis, this corresponds to strengthen the adversary by avoiding the perturbation of these zero-coefficients.

For the expression $w^T x$, let Z be the set of indices i with w_i fixed to zero. Let $\mathcal{S}$ be the set of solutions. We call two solutions $x_1, x_2 \in \mathcal{S}$ equivalent, if they differ only in variables from Z. This way, Z defines equivalence classes on $\mathcal{S}$ with respect to the expression $w^T x$. Obviously, $w^T x$ evaluates to the same value for solutions in the same equivalence class.

In the following we show that the Separation Lemma is unaffected by zero preserving perturbations. Assume we fix $n-l$ coefficients to zero and, without loss of generality, these are the last coefficients, i.e., $Z = \{l+1, l+2, \ldots, n\}$. Hence, $w = (w_1, \ldots, w_l, 0, \ldots, 0)$. Recall that solutions are $0/1$ vectors of length n. Solutions that agree in the first l positions are equivalent. Observe that only the highest ranked solution in each equivalence class is relevant for the loser and feasibility gap. For the purpose of the analysis, we virtually remove all other solutions. For convenience, let us call the resulting solution set $\mathcal{S}$. In order to apply the Separating Lemma, we define a reduced problem with l variables corresponding to the first l variables of the original problem. The reduced solution set $\mathcal{S}'$ is obtained accordingly by cutting the last $n-l$ entries of each solution in $\mathcal{S}$. Notice that $x \in \mathcal{S}$ and the corresponding $x' \in \mathcal{S}'$ have the same weight $w^T x$. The rank of a $x' \in \mathcal{S}'$ is the rank of the corresponding solution $x \in \mathcal{S}$. This way, solution x is a loser (winner) if and only if the corresponding x' is a loser (winner). Hence, the feasibility gap as well as the loser gap are identical in the original and the reduced problem.

A similar argument can be used to show that the Generalized Isolating Lemma stays valid with respect to equivalence classes, that is, the winner gap is defined to be the difference in profit between the best and the second best equivalence class. However, it might be very likely that there are many optimal solutions, as the winning equivalence class might have many solutions. Notice that this effects the procedure that verifies optimality for stochastic objective functions, described in the proof of Theorem 12. In particular, the two solutions x' and x'', which are optimal with respect to the rounded cost vectors $\lfloor c \rfloor$ and $\bar{c}$, only have to be in the same equivalence class to certify optimality, which can be checked easily.

6.1 Algorithmic Application

Using zero-preserving perturbations, we can apply our analysis to the General Assignment Problem (GAP), which is defined as follows.

$$\begin{array}{llll} \max & \left(\sum_{i=1}^{k}\sum_{j=1}^{n} p_{ij}x_{ij}\right) \quad \text{subject to} & \sum_{j=1}^{n} w_{ij}x_{ij} \le c_i, & i = 1,\ldots,k \\ & & \sum_{i=1}^{k} x_{ij} = 1, & j = 1,\ldots,n \\ & & x_{ij} \in \{0,1\}, & \text{for all } i \in [k], j \in [n] \enspace . \end{array}$$

Intuitively, the goal is to pack n items into k bins with capacities $c_1, \ldots, c_k$. Packing item $j \in [n]$ into bin $i \in [k]$ occupies w_{ij} units of the capacity of bin i and yields a profit of p_{ij}. Even for two bins, the GAP-problem does not allow an FPTAS, unless $\mathsf{P} = \mathsf{NP}$ [18]. In fact, the problem is strongly NP-hard for an unbounded number of bins. If however, the number of bins is a constant, then GAP can be solved in pseudo-polynomial time using standard dynamic programming as follows. Let

$T(j, W_1, \ldots, W_k)$ denote the maximum profit over solutions that use only items $\{1, \ldots, j\}$ such that the cumulative weight of all items in bin i is exactly W_i, for all $i \in [k]$. All non-trivial entries can be computed by the following recursion.

$$T(j, W_1, \ldots, W_k) = \max_{i \in [k]} \{T(j-1, V_1, \ldots, V_k) + p_{ij} \mid V_i = W_i - w_{ij}, V_l = W_l, \text{ for all } l \neq i\}$$

Let W denote the largest weight of the instance, i.e., $W = \max_{j \in [n]} \max_{i \in [k]} w_{ij}$. Then the size of the dynamic programming table is at most $n(nW)^k$. Hence, the running time is at most $kn(nW)^k$.

How does the GAP problem fit into our framework? We have a vector of kn binary variables $(x_{11}, x_{12}, \ldots, x_{1n}, x_{21}, x_{22}, \ldots, x_{2n}, \ldots, x_{k1}, x_{k2}, \ldots x_{kn})$ and the corresponding vector of profits. The weight constraint for bin $l \in [k]$ uses the weight vector with all entries w_{ij}, $i \neq l$, set to zero. These entries are declared to be fixed to zero, so they are not touched by the perturbation. This way, each w_{ij} appears in just one constraint and a perturbation of the profit and the constraint vectors has the same effect as perturbing each p_{ij} and w_{ij} individually. This yields

Corollary 15 *The General Assignment Problem with a constant number of constraints/bins has polynomial smoothed complexity if weights and profits are randomly perturbed.*

The discussion above reveals a limitation of our analysis. For this purpose, consider the multiple knapsack problem, a special case of the GAP problem, where the weight of an item is the same for all bins, i.e., $w_{ij} = w_j$, for all $i \in [k]$. Using the program formulation of the GAP problem, the weights w_j reoccur in the different constraints for the different knapsacks. The perturbation of the constraints, however, have to be independent such that the perturbed instance is, in general, no multiple knapsack instance any more. We observe that for a sensible application of our analysis, each variable that describes the problem instance, must appear at most once in the stochastic constraints of the linear program formulation. Or, seeing it another way, the coefficients in the stochastic constraints must allow independent perturbations without destroying the structure of the problem.

7 Other Aspects

7.1 Smoothed complexity and approximation schemes

If a binary optimization problem Π has polynomial smoothed complexity when perturbing only the coefficients in the objective function, then Π also admits an absolute fully polynomial randomized approximation scheme. The idea is to perturb the instance only slightly, resulting in a very similar optimal objective function value. The set of feasible solutions is not affected by the perturbation. In order to bound the running time, we can exploit the positive influence of the added randomness. Let us give some more details.

Given a (worst-case) instance of Π, the coefficients in the objective function are normalized such that the largest absolute value is 1. Subsequently, the normalized coefficients are perturbed using the uniform perturbation model with parameter ϕ. As each coefficient changes by at most $1/\phi$, the difference in the objective values of any two solutions can change by at most n/ϕ by this perturbation. Hence, the optimal solution of the perturbed instance is, with respect to the unperturbed objective function, at most n/ϕ away from the optimum. According to the definition of polynomial smoothed complexity, we can fix some polynomial bound B such that the running time of the algorithm is smaller than B with probability at least $1/2$. In order to obtain polynomial expected running time we generate perturbations and run the algorithm for at most B time steps. If the algorithm has not finished, we generate a new perturbation and try again. As the perturbations are independent, one needs 2 iterations on average until a solution is found. Hence, the expected running time is $2B$.

The approximation scheme only allows to bound the absolute error. If however, the optimum of the normalized instance can be lower bounded by some polynomial in $1/n$, then we obtain a randomized FPTAS.

The opposite direction works as well. Suppose Π is a binary optimization problem admitting a fully polynomial approximation scheme. It is well known that such an approximation scheme can be transformed into a pseudo-polynomial algorithm, cf. [10]. The running time of this algorithm is pseudo-polynomial only with respect to the coefficients in the objective function. Thus, Theorem 12 shows that Π has polynomial smoothed complexity when perturbing only the objective function.

This shows that for the considered class of problems, there exists a correlation between problems that allow an FPTAS and problems with polynomial smoothed complexity for stochastic linear objective functions.

Observe that these results rely solely on the analysis of the winner gap. The results based on the loser and feasibility gaps yield polynomial smoothed complexity for problems for which no fully polynomial approximation scheme is known like, e.g., constrained spanning tree, or even for problems that cannot be approximated within any polynomial-time computable factor like, e.g., 0/1 programming with a constant number of constraints. In fact, the problems to which these gaps apply are exactly those problems admitting a bicriteria approximation scheme. In other words, our results show that the impact of randomly perturbing the objective function and/or the constraints on the computational complexity of a problem is comparable with the impact of relaxing the same set of expressions. The advantage of smoothed analysis against (bicriteria) approximation schemes, however, is that it yields optimal and feasible instead of almost optimal and almost feasible solutions.

7.2 Euclidean optimization problems.

On a first view, the Euclidean variants of TSP and Steiner tree might look like interesting candidates for problems with polynomial smoothed complexity. Karp [13] in a seminal work on the probabilistic analysis of algorithms studied the TSP problem in a model in which n points are drawn uniformly and independently from the unit square. A natural extension of this model would be to assume that first an adversary chooses points in the unit cube or ball and then one applies a multi-dimensional Gaussian perturbation. We claim, however, that neither Euclidean TSP nor Steiner tree do have polynomial smoothed complexity, since the perturbation does not change the set of feasible solutions. The change in the objective function due to the perturbation depends at most linearly on the magnitude of the perturbation. Since the optimal objective value is $\Omega(1)$, we could obtain a fully polynomial randomized approximation scheme for worst-case instances, which is widely believed not to exist. Thus, one needs to give up either the requirement that the running time is polynomial in $1/\varepsilon$ or the requirement that the running time is polynomial in ϕ. Under the first relaxation the problem has been solved by Arora [1]. The question whether there exist polynomial time algorithms for Euclidean TSP or Steiner tree under perturbations with a fixed density parameter ϕ or in the uniform input model of Karp [13] is open.

7.3 Relationship to condition numbers

In order to obtain a finer analysis of algorithms than that provided by worst-case complexity, one should find a way of distinguishing hard problem instances from easy ones. A natural approach is to find a quantity indicating the difficulty of solving a problem instance. In Numerical Analysis and Operations Research it is common to bound the running time of an algorithm in terms of a *condition number* of its input. The condition number is typically defined to be the sensitivity of the solution for a problem instance to slight perturbations of the input. For example, Renegar [23, 24, 25] presents a variant of the primal interior point method and describes its running time as a function of the condition number. Remarkably, his running time bound depends only logarithmically on the condition number. Dunagan, Spielman, and Teng [8] study this condition number in the smoothed analysis framework. Assuming Gaussian ϕ-perturbations, the condition number can be bounded by a function that is polynomial in ϕ. Thus, the running time of Renegar's algorithm depends only logarithmically on the density parameter ϕ. In contrast, the running time bound of the Simplex algorithm presented by Spielman and Teng in [27] is polynomial in ϕ.

In [28], Spielman and Teng propose to extend the condition number towards discrete optimization problems in order to assist the smoothed analysis of such problems. As a natural definition for the condition number of a discrete function they suggest *the reciprocal of the minimum distance of an input to one on which the function has a different value*. In fact, the minimum of winner, loser, and feasibility gap is a lower bound on the amount by which the coefficients of a binary optimization problem need to be altered so that the winner, i.e., the solution taking the optimal value, changes. Let us define the reciprocal of this minimum to be the *condition number for binary optimization problems*. This allows us to summarize our analysis in an alternative way. Our probabilistic analysis in Section 2 shows that the condition number is bounded polynomially in the density parameter ϕ. Furthermore, in Section 3, we proved that a problem with pseudo-polynomial worst-case complexity admits an algorithm whose running time is bounded polynomially in the condition number. Combining these results, we obtained algorithms whose smoothed complexity depends in a polynomial fashion on the density parameter ϕ. Let us remark that this kind of dependence is best possible for NP-hard optimization problems, unless there is a subexponential time algorithm for NP-complete problems. In particular, a running time bound logarithmic in ϕ would yield a randomized algorithm with polynomial worst-case complexity.

References

[1] S. Arora, *Polynomial time approximation schemes for euclidean TSP and other geometric problems*, in IEEE Symposium on Foundations of Computer Science (FOCS), 1996, pp. 2–11.

[2] C. Banderier, R. Beier, and K. Mehlhorn, *Smoothed analysis of three combinatorial problems*, in Proc. 28th International Symposium on Mathematical Foundations of Computer Science (MFCS-2003), vol. 2747, Bratislava, Slovak Republic, 2003, pp. 198–207.

[3] F. Barahona and W. R. Pulleyblank, *Exact arborescences, matchings and cycles*, Discrete Applied Mathematics, 16 (1998), pp. 617–630.

[4] L. Becchetti, S. Leonardi, A. Marchetti-Spaccamela, G. Schäfer, and T. Vredeveld, *Average case and smoothed competitive analysis of the multi-level feedback algorithm*, in Proceedings of the Forty-Fourth Annual IEEE Symposium on Foundations of Computer Science, 2003, pp. 462–471.

[5] R. Beier and B. Vöcking, *Random knapsack in expected polynomial time*, in Proc. 35th Annual ACM Symposium on Theory of Computing (STOC-2003), San Diego, USA, 2003, pp. 232–241.

[6] ——, *Probabilistic analysis of knapsack core algorithms*, in Proc. 15th Annual Symposium on Discrete Algorithms (SODA-2004), New Orleans, USA, 2004, pp. 461–470.

[7] V. Damerow, F. M. auf der Heide, H. Räcke, C. Scheideler, and C. Sohler, *Smoothed motion complexity*, in Proceedings of the 11th European Symposium on Algorithms, 2003, pp. 161–171.

[8] J. Dunagan, D. Spielman, and S.-H. Teng, *Smoothed analysis of interior-point algorithms: Condition number*, Math. Programming, (2003).

[9] M. DYER AND A. M. FRIEZE, *Probabilistic analysis of the multidimensional knapsack problem*, Maths. of Operations Research, 14(1) (1989), pp. 162–176.

[10] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability*, Freeman and Company, New York, 1979.

[11] M. X. GOEMANS AND R. RAVI, *The constrained minimum spanning tree problem*, in Fifth Scandinavian Workshop on Algorithm Theory, LNCS 1097, Springer, 1996, pp. 66–75.

[12] A. GOLDBERG AND A. MARCHETTI-SPACCAMELA, *On finding the exact solution to a zero-one knapsack problem*, in Proceedings of the 16th ACM Symposium on Theory of Computing (STOC), 1984, pp. 359–368.

[13] R. M. KARP, *Probabilistic analysis of partitioning algorithms for the traveling salesman problem*, Oper. Res, 2 (1977), pp. 209–224.

[14] H. KELLERER, U. PFERSCHY, AND D. PISINGER, *Knapsack Problems*, Springer, 2004.

[15] G. S. LUEKER, *On the average difference between the solutions to linear and integer knapsack problems*, Applied Probability - Computer Science, the Interface, Birkhäuser, 1 (1982), pp. 489–504.

[16] ——, *Average-case analysis of off-line and on-line knapsack problems*, Journal of Algorithms, 19 (1998), pp. 277–305.

[17] ——, *Exponentially small bounds on the expected optimum of the partition and subset sum problems*, Random Structures and Algorithms, 12 (1998), pp. 51–62.

[18] S. MARTELLO AND P. TOTH, *Knapsack Problems – Algorithms and Computer Implementations*, Wiley, 1990.

[19] K. MEHLHORN AND M. ZIEGELMANN, *Resource constrained shortest paths*, in Proc. 8th Annual European Symp. (ESA-00), LNCS 1879, Springer, 2000, pp. 326–337.

[20] K. MULMULEY, U. VAZIRANI, AND V. VAZIRANI, *Matching is as easy as matrix inversion*, Combinatorica, 7(1) (1987), pp. 105–113.

[21] C. H. PAPADIMITRIOU AND M. YANNAKAKIS, *The complexity of tradeoffs and optimal access of web sources*, in IEEE Symposium on Foundations of Computer Science (FOCS), 2000, pp. 86–92.

[22] R. G. PARKER, *Deterministic scheduling theory*, Chapman & Hall, 1995.

[23] J. RENEGAR, *Some perturbation theory for linear programming*, Math. Programming, 65(1, Series A) (1994), pp. 73–91.

[24] ——, *Incorporating condition measures into the complexity of linear programming*, SIAM J. Optimization, 5(3) (1995), pp. 506–524.

[25] ——, *Linear programming, complexity theory and elementary functional analysis*, Math. Programming, 70(3, Series A) (1995), pp. 279–351.

[26] H. M. SALKIN, *Integer Programming*, Addison-Wesley, 1975.

[27] D. A. SPIELMAN AND S.-H. TENG, *Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time*, in Proceedings of the 33rd ACM Symposium on Theory of Computing (STOC), 2001, pp. 296–305.

[28] ——, *Smoothed analysis: Motivation and discrete models*, in Proceedings of WADS, 2003.