Angewandte Mathematik und Informatik
Universität zu Köln

Report No. 93.147

Algorithmic Chernoff-Hoeffding Inequalities in Integer Programming

by

Anand Srivastav[1], Peter Stangier

1995

Random Structures and Algorithms January 1996 and
Proceedings of the 5th International Annual Conference on Algorithms and Computation, ISAAC'94, Beijing, Lecture Notes in Computer Science, Vol. 834, pages 226 – 234, Springer Verlag.

Anand Srivastav

Institut für Informatik — Lehrstuhl Algorithmen und Komplexität
Humboldt Universität zu Berlin
Unter den Linden 6
10099 Berlin
Germany
email: srivasta@informatik.hu-berlin.de

Peter Stangier

Institut für Informatik
Universität zu Köln
Pohligstr.1
50969 Köln
Germany
stangier@informatik.uni-koeln.de

[1]supported by: Deutsche Forschungs Gemeinschaft

1991 Mathematics Subject Classification: 60C05 60E15 68Q25 90C10 90C27 90B35 90C35
Keywords: randomized algorithms, derandomization, approximation algorithms, integer programming, resource constrained scheduling

Algorithmic Chernoff-Hoeffding Inequalities in Integer Programming

Anand Srivastav Peter Stangier

September 1995

Abstract

Proofs of classical Chernoff-Hoeffding bounds has been used to obtain polynomial-time implementations of Spencer's derandomization method of conditional probabilities on usual finite machine models: given m events whose complements are large deviations corresponding to weighted sums of n mutually independent Bernoulli trials, Raghavan's lattice approximation algorithm constructs for $0-1$ weights and integer deviation terms in $O(mn)$-time a point for which all events hold. For rational weighted sums of Bernoulli trials the lattice approximation algorithm or Spencer's hyperbolic cosine algorithm are deterministic precedures, but a polynomial-time implementation was not known. We resolve this problem with an $O(mn^2 \log \frac{mn}{\epsilon})$-time algorithm, whenever the probability that all events hold is at least $\epsilon > 0$. Since such algorithms simulate the proof of the underlying large deviation inequality in a constructive way, we call it the algorithmic version of the inequality. Applications to general packing integer programs and resource constrained scheduling result in tight and polynomial-time approximations algorithms.

1 Introduction

In many applications of the probabilistic method combinatorial structures can be represented as a collection of events $E_1, \ldots, E_m$, whose complements E_i^c describe large deviations in a finite probability space: for $i = 1, \ldots, m$ and $j = 1, \ldots, n$ let (w_{ij}) be a $m \times n$ matrix with $w_{ij} \in [0,1] \cap \mathbb{Q}$. Let $X_1, \ldots, X_n$ be mutually independent $0-1$ random variables with rational expectation $\mathbb{E}(X_j) = p_j$ and let ψ_i be the weighted sums $\psi_i = \sum_{j=1}^n w_{ij} X_j$. Given rational deviation parameters $\lambda_i > 0$, denote by E_i exactly one of the events

$$\text{“}\psi_i \leq E(\psi_i) + \lambda_i\text{”} \ or \ \text{“}\psi_i \geq E(\psi_i) - \lambda_i\text{”},$$

$i = 1, \ldots, m$. The various types of Chernoff-Hoeffding bounds for $\mathbb{P}(E_i^c)$ can be summarized by the inequalites

$$\mathbb{P}(E_i^c) \leq e^{-t_i \lambda_i} \mathbb{E}(e^{t_i \psi_i}) \leq f(\lambda_i), \tag{1}$$

where an optimal choice of the parameter $t_i > 0$ gives the sharpest possible upper bound $f(\lambda_i)$ and f is a function exponentially decaying in λ_i. If $\sum_{i=1}^m f(\lambda_i) < 1 - \epsilon$ for some $0 < \epsilon < 1$, then $\mathbb{P}(\bigcap_{i=1}^m E_i) \geq \epsilon$, hence $\bigcap_{i=1}^m E_i$ is not empty and derandomization is the task of constructing a point in $\bigcap_{i=1}^m E_i$ in polynomial-time. In principle this can be done by the conditional probability method due to Spencer (see also Erdös/ Selfridge [11]), for example with Spencer's hyperbolic cosine algorithm [26] or Raghavan's lattice approximation algorithm [23]. But the efficiency of these algorithms heavily depends on the efficient computation of the conditional probabilities or of appropriate upper bounds on them on finite machine models, like the usual RAM or Turing machine model. In particular, the computation of the moment generating functions $\mathbb{E}(e^{t_i \psi_i})$ is required. This indeed is possible in the following cases:

- For $0-1$ weighted sums of Bernoulli trials and integer λ_i Raghavan's lattice approximation algorithm has an $O(nm)$-time implementation on the RAM model and can be considered as an algorithmic form of the Raghavan/Spencer bound ([23], Theorem 1 and 2) (which is slightly weaker than the Angluin-Valiant inequality ([18], Corollary 5.2 (b)).

- For $0-1$ weighted sums of Bernoulli trials, uniform distribution, $n = m$ and $\lambda_i = \lceil n^{\frac{1}{2}+\delta}\sqrt{\ln 2n}\rceil$ $(0 < \delta < 1)$ a NC algorithms is known: one can use either the method of $(\log^c n)$-wise independence (Beger,Rompel [5], Motwani, Naor, Naor [20]) or the construction of small bias probability spaces (Naor, Naor [21]) to design a parallel $O(\log^3 n)$-time algorithm for the construction of a point in $\bigcap_{i=1}^m E_i$ using $O(n^{3+\frac{1}{\delta}})$ PRAM processors. Sequentially implemented this gives the running time of $O(n^{3+\frac{1}{\delta}} \log^3 n)$.

Unfortunately, for rational weights w_{ij} and optimal choice of t_i the moment generating functions $\mathbb{E}(e^{t_i\psi_i})$ necessarily are transcendental, therefore cannot be exactly computed on a finite machine model, which on the other hand is presumed for a polynomial running time of the conditional probability method. Of course, if we neglect computational errors, for example using floating point arithmetics, the conditional probability method runs in $O(nm)$-time, no matter what the parameters or weights are. But from the computational complexity point of view, when the underlying computational model is a Turing machine or the RAM model, floating point arithmetics is not satisfactory:

- The correctness of the algorithm is in doubt, when approximations are done without provable guarantees.

- The cost of numerical approximations is a part of the total running time, consequently has to be taken into acount.

Indeed, Feldstein and Turner [8] confirmed in theoretical models that floating point arithmetic can cause loss of significance. In conclusion, we have to insist on exact computations. For a comprehensive discussion of the advantages of the exact computation paradigm versus floating point arithmetic we refer to the recent paper of C-K. Yap [31].

For rational weighted sums of Bernoulli trials it remained an open problem, whether the conditional probability method has a polynomial-time implementation on usual models of computation, like the RAM model or the Turing machine model (remark on page 138 in [23]).

As a main result of this paper we resolve this problem for various bounds from the Chernoff-Hoeffding family and obtain results of the following form.

Let $0 < \epsilon < 1$. Whenever $\mathbb{P}(\bigcup_{i=1}^m E_i^c) \leq \sum_{i=1}^m f(\lambda_i) < 1-\epsilon$, then a point in $\bigcap_{i=1}^m E_i$ can be constructed in $O(mn^2 \log \frac{mn}{\epsilon})$-time.

The algorithm behind this result gives a clear and unified implementation of the conditional probability method and since it heavily simulates the proof of the underlying large deviation bound, we call it the algorithmic version of the inequality under consideration.

For a fix success probability $\epsilon > 0$ we have a strongly polynomial algorithm, i.e. an algorithm with running time independent of the - perhaps large - encoding length of the numbers w_{ij}, λ_i, p_j appearing in the problem. This has an important consequence in applications to integer programming, where the randomized rounding/derandomization scheme is applied. In the randomized rounding step an optimal solution to the linear programming relaxation is generated. This solution draws a probability distribution and helps to derive with non-zero probability a good approximation of the integral optimum. In the second step derandomization constructs such an approximation. Since for many LP's fast, strongly polynomial-time algorithms are known, for example the Tardos' algorithm [30], it is desirable to combine them with a strongly polynomial derandomization procedure in order to derive strongly polynomial approximation algorithms. We show the following two applications of algorithmic versions of Chernoff-Hoeffding bounds:

Consider the packing integer program

$$\max\{c^T x; Ax \leq b\}$$

with $c \in [0,1]^n$, $a_{ij} \in [0,1] \cap \mathbb{Q}$ and $x \in \mathbb{N}^n$. In the case of $0-1$ variables x_i, $0-1$ entries a_{ij}, $c_i = 1$ and $b_i = k$ for some constant integer k, Raghavan's [23] hypergraph k-matching algorithm gives an approximation of the integer maximum within a factor of $1 - D(k,m,n)$. For $k \geq \ln m$ the function $D(k,m,n)$ is constant, thus a constant factor approximation is achieved. We cover the integer problem in its full generality and show for every $0 < \epsilon \leq \frac{9}{10}$ and instances with not too small packing constraints b_i, i.e. $b_i = \Omega(\frac{1}{\epsilon^2}\log m)$, an $(1-\epsilon)$-approximation of the integer optimum in deterministic polynomial-time. In particular a randomized rounding technique is introduced, which removes Raghavan's restriction to $0-1$ integer programs.

Furthermore we consider a classical resource constrained scheduling problem, where the makespan has to be minimized ([13], problem SS10, p. 239). We present the first 2-factor approximation algorithm and prove that the factor 2 is nearly optimal. In particular, a reduction of the scheduling problem to the problem of partitioning a graph into 2 perfect matchings proves for every $\rho < 1.5$ that the existence of a polynomial-time ρ-approximation algorithm would imply P = NP.

The algorithmic Chernoff-Hoeffding inequalities derived in this paper constitute basic derandomization tools, and has been applied to some other packing integer programs: In [27] a more sophisticated analysis of the algorithmic Angluin-Valiant bound in the special case of maximal weighted k-matching in hypergraphs results in a faster derandomization procedure for this problem. A direct application of the approximation algorithm for integer programming presented in this paper to the hypergraph k-matching problem would require a derandomization time of $O(mn^2 \log \frac{mn}{\epsilon})$, while in [27] the improved running time of $O(mn + n^2 \log n)$ is shown. For the feasibility multicommodity flow problem good deterministic approximation algorithms along with non-approximability proofs are given in [28] and more about approximability/non-approximability of resource constrained scheduling can be found in [29].

In this paper we consider the RAM model with unit cost [19] for multiplication and distinguish between polynomial and strongly polynomial algorithms, defined in the usual way: By the size of an input we mean the number of data items in the descripton of the input, while the encoding length of the input is the maximal binary encoding length of data items in the input. On the RAM model an algorithm runs in polynomial-time (resp. strongly polynomial-time), if the number of elementary arithmetic operations (briefly called running time) is polynomially bounded in both the size and the encoding length of the input (resp. *only* in the size of the input) *and* in addition the maximal binary encoding length of a number appearing during the execution of the algorithm (briefly called space) is polynomially bounded in the size and encoding length of the input.

Note that all so defined polynomial-time algorithms are also polynomial-time algorithms on the Turing machine model, because we require that the encoding length of numbers is polynomially bounded in the input size. This is *not* the case in "pure" RAM models, where one only counts elementary arithmetic operations, regardless of the size of numbers.

2 Algorithmic Chernoff-Hoeffding Type Inequalities

In the following subsection we cite the basic inequalities, whose algorithmic counterpart we wish to derive.

2.1 Chernoff-Hoeffding Type Inequalities

Let $X_1, \ldots, X_n$ be mutually independent random variables, where X_j is equal to an integer u_j with probability p_j and is equal to an other integer v_j with probability $1 - p_j$. For $1 \leq j \leq n$ let

w_j denote rational weights with $0 \le w_j \le 1$ and denote by ψ the random variable

$$\psi = \sum_{j=1}^{n} w_j X_j.$$

A basic large deviation inequality is due to Bernstein (see [10]) and Chernoff [9] in the Binomial case ($u_j = 1, v_j = 0$, $p_j = p, w_j = 1$ for all $j = 1, \ldots, n$) and has been generalized by Hoeffding [15]:

Theorem 2.1 *(Bernstein, Chernoff, Hoeffding) Let $u_j = 1, v_j = 0$ for all $j = 1, \ldots, n$ and let $\lambda > 0$. Then*

(a) $\mathbb{P}(\psi > \mathbb{E}(\psi) + \lambda) \le \exp(-\frac{2\lambda^2}{n})$

(b) $\mathbb{P}(\psi < \mathbb{E}(\psi) - \lambda) \le \exp(-\frac{2\lambda^2}{n})$.

In the literature Theorem 2.1 is well known as the Chernoff bound. For k-wise independent random variables similar bounds can be found in the recent paper of Schmidt, Siegel and Srinivasan [25].

For small expectations, i.e $\mathbb{E}(\psi) \le \frac{n}{6}$, the following inequalities, which have been attributed to Angluin and Valiant [3], give sharper bounds.

Theorem 2.2 *(Angluin and Valiant) Let $u_j = 1, v_j = 0$ for all $j = 1, \ldots, n$ and let $0 < \beta \le 1$. Then*

(a) $\mathbb{P}(\psi > \mathbb{E}(\psi)(1 + \beta)) \le \exp(-\frac{\beta^2 \mathbb{E}(\psi)}{3})$

(b) $\mathbb{P}(\psi < \mathbb{E}(\psi)(1 - \beta)) \le \exp(-\frac{\beta^2 \mathbb{E}(\psi)}{2})$.

For random variables with zero expectation there are two inequalities which can be found in the book of Alon and Spencer ([1] , Appendix). The first inequality goes back to Hoeffding, while the second inequality is due to Alon and Spencer [1]. The proof of Alon and Spencer requires $w_j = 1$ for all j, but an examination of their arguments shows that w_j can be $0 - 1$.

Theorem 2.3 *(Hoeffding) Let $u_j = 1 - p_j, v_j = -p_j$, $w_j \in \{0, 1\}$ for all $j = 1, \ldots, n$ and let $\lambda > 0$. Then*

(a) $\mathbb{P}(\psi > \lambda) \le \exp(-\frac{2\lambda^2}{n})$

(b) $\mathbb{P}(\psi < -\lambda) \le \exp(-\frac{2\lambda^2}{n})$.

Alon and Spencer improved the Hoeffding bound $e^{-\frac{2\lambda^2}{n}}$ replacing n by $pn = p_1 + \ldots + p_n$ which is an upper bound for $\mathrm{Var}(\psi)$.

Theorem 2.4 *(Alon, Spencer) Let $u_j = 1 - p_j, v_j = -p_j$, $w_j = 1$ for all $j = 1, \ldots, n$ and let $\lambda > 0$. Set $p = \frac{1}{n}(p_1 + \ldots + p_n)$. Then*

(a) $\mathbb{P}(\psi > \lambda) \le \exp(-\frac{\lambda^2}{2pn} + \frac{\lambda^3}{2(pn)^2})$

(b) $\mathbb{P}(\psi < -\lambda) \le \exp(-\frac{\lambda^2}{2pn})$.

$\square$

In the next section we prepare the technical tools for the approximate computation of conditional probabilities and moment generating functions for weighted sums of Bernoulli trials.

2.2 Pessimistic Estimators and Elementary Functions

Let us start with a definition of the derandomization problem. Let $(\Omega, \mathbb{P})$ be a probability space, and for simplicity assume that Ω is the set of all vectors of length n with components from a finite set S. Let $E_1, \ldots, E_m$ be a collection of events such that $\mathbb{P}(\bigcap_{i=1}^m E_i) \geq \epsilon$ for some $0 < \epsilon < 1$.

Definition 2.5 *(Derandomization Problem) Find a vector* $x \in \bigcap_{i=1}^m E_i$ *in deterministic time bounded by a polynomial in* $n, m, |S|$ *and* $\log \frac{1}{\epsilon}$.

The "conditional probability method" is the following algorithm:

Algorithm CONDPROB

INPUT: An event $E \subset \Omega$ with $\mathbb{P}(E) > 0$.

OUTPUT: A vector $x \in E$.

1. Choose x_1 as the miminizer of the function $\omega \mapsto \mathbb{P}[E^c|\omega]$, $\omega \in S$.
For $l = 2, \ldots, n$ do:

If $x_1 \ldots, x_{l-1}$ with $x_i \in S$ have been selected, set x_l where x_l minimizes the function $\omega \mapsto \mathbb{P}[E^c|x_1 \ldots, x_{l-1}, \omega]$, $\omega \in S$.

□

The striking observation is that a so constructed x satisfies $x \in E$. But it is hard to compute conditional probabilities directly. Spencer's hyperbolic cosine algorithm [26] shows that this is not really necessary, if upper bounds on conditional probabilities can be computed which behave like conditional probabilities. This fact has been conceptualized by Raghavan [23] who introduced the notion of "pessimistic estimators".

Definition 2.6 *(Pessimistic Estimator, [23]) Let* $(\Omega, \mathbb{P})$ *be a probability space as defined above. Let* $E_1, \ldots, E_m$ *be a collection of events and let* E *denote the event* $\bigcap E_i$. *Suppose that* $\mathbb{P}(E) > \epsilon$, $\epsilon > 0$. *A pessimistic estimator for the event* E^c *is a sequence* $(U_l^{min}(x_1, \ldots, x_l))_{l=1}^n$ *which iteratively construct a vector* $(x_1, \ldots, x_n) \in \Omega$ *and possess the following properties for all* $1 \leq l \leq n$:

(a) $\mathbb{P}(\bigcup_{i=1}^m E_i^c | x_1, \ldots, x_l) \leq U_l^{min}(x_1, \ldots x_l)$

(b) $U_{l+1}^{min}(x_1, \ldots, x_l, x_{l+1}) \leq U_l^{min}(x_1, \ldots, x_l)$

(c) $U_1^{min}(x_1) < 1$

(d) Each $U_l^{min}(x_1, \ldots, x_l)$ *can be computed in time bounded by a polynomial in* $n, m, |S|$ *and* $\log(1/\epsilon)$.

Given a pessimistic estimator $x = (x_1, \ldots, x_n)$ is the desired vector, because the conditions (a), (b) and (c) imply:

$$\mathbb{P}(\bigcup_{i=1}^m E_i^c | x_1, \ldots, x_n) < 1,$$

hence

$$\mathbb{P}(\bigcup_{i=1}^m E_i^c | x_1, \ldots, x_n) = 0,$$

therefore $x \in \bigcap_{i=1}^m E_i$.

By Definition 2.6 upper bounds on conditional probabilities are the potential candidates for pessimistic estimators. Since in case of sums of independent random variables such upper bounds typically are compositions of elementary functions, we need to compute them, at least in an approximate fashion. Lemma 2.7 shows that an approximate computation of elementary functions like $\exp(z), \ln(z)$ and $\sqrt{z}$ can be done efficiently. It is related to Brent's [7] approximation of elementary functions defined over compact intervals, but the advantage of our approximation is that we can deal with arbitrarily large rational numbers. Lemma 2.8 shows that a product of compositions of exponential functions and logarithms is efficiently approximable and Lemma 2.9 is a simple observation which will be used to prove the decreasing monotonicity of the pessimistic estimator.

Lemma 2.7 *(i) Let y be a rational number with encoding length L and let $\gamma_1 \in (0,1)$ be a positive real number. Let N be a positive integer with $N \geq 8\lceil |y| \rceil + \lceil \log \frac{1}{\gamma_1} \rceil$. Then the N-th degree Taylor polynomial*
$T_N(y) = \sum_{k=0}^{N} \frac{y^k}{k!}$ of $\exp(y)$ has encoding length $O(LN + N \log N)$, can be computed in $O(N)$ time and the inequality $|\exp(y) - T_N(y)| \leq \gamma_1$ holds.

(ii) Let $x \geq 1$ be a rational number, $\gamma_2 \in (0,1)$ a real number and $L_0 = \lfloor \log x \rfloor$. For every $N \geq \lceil \log \frac{4L_0}{\gamma_2} \rceil$ a rational number y with encoding length $O(LN)$ can be computed in $O(L_0 + N)$ time such that $|\ln\ x - y| \leq \gamma_2$.

(iii) Let x be a rational number with encoding length L, $\gamma_3 \in (0,1)$ a positive real number. If $x \geq 1$, then let N be a positive integer with $N \geq \lceil \log \frac{x}{\gamma_3} \rceil$ and if $0 < x < 1$, then suppose that $N \geq \lceil \log \frac{1}{\gamma_3} \rceil$. A rational number y with encoding length $O(L + N)$ can be computed in $O(N)-$time such that $|\sqrt{x} - y| \leq \gamma_3$.

Proof.
(i) Since $N \geq 3|y|$ we have by Taylor's theorem

$$|\exp(y) - T_N(y)| \leq \frac{|y|^{N+1}}{(N+1)!} \leq \frac{|y|^N}{N!},$$

and observing that $N \geq e^2|y|$, $N! \geq (\frac{N}{e})^N$ and $N \geq \ln \frac{1}{\gamma_2}$ the inequality follows. Since $\frac{y^{i+1}}{(i+1)!}$ is calculated from $\frac{y^i}{i!}$ in constant time, $T_N(y)$ is computed in $O(N)-$time. Furthermore the encoding length of $T_N(y)$ is a polynomial in L and N : The encoding length of y^N is $O(LN)$, $N!$ has encoding length $\Theta(N \log N)$ and $T_N(y)$ has encoding length $O(LN + N \log N + N) = O(LN + N \log N)$.

(ii) For the computation of $\ln x$ we use its power series expansion. With $L_0 = \lfloor \log x \rfloor$ as in the lemma, we have $2^{L_0} \leq x \leq 2^{L_0+1}$, and we can find L_0 in $O(L_0)$-time. Define

$$y_0 = \begin{cases} x2^{-L_0} & \text{if } 1 \leq x2^{-L_0} \leq 1.5 \\ \frac{3}{4}x2^{-L_0} & \text{if } 1.5 < x2^{-L_0} \leq 2 \end{cases}$$

and use the decomposition $x = 2^{L_0}y_0$ or $x = \frac{4}{3}2^{L_0}y_0$. It is enough to consider the second case $x = \frac{4}{3}2^{L_0}y_0$, because the arguments in the other case are the same.
There exists a rational number y_1, $0 < y_1 \leq \frac{1}{2}$ with $y_0 = 1 + y_1$, and we have the decomposition

$$\ln\ x \quad = \quad L_0[\ln\,(1 + \frac{1}{3}) + \ln\,(1 + \frac{1}{2})] + \ln\,(1 + \frac{1}{3}) + \ln\,(1 + y_1).$$

Let $S_J(z) := \sum_{j=1}^{J} (-1)^{j-1} \frac{z^j}{j}$. Then with $J_1 = \lceil \log(\frac{4L_0}{\gamma_2}) \rceil - 1$ and $0 < y_1 \leq \frac{1}{2}$ we get

$$|\ln\,(1 + y_1) - S_{J_1}(y_1)| \leq \frac{y_1^{J_1+1}}{J_1 + 1} \leq \frac{1}{2^{J_1+1}} \leq \frac{\gamma_2}{4L_0}.$$

Choosing $J_2 = \lceil \log_3(\frac{4}{\gamma_2}) \rceil - 1$, $J_3 = \lceil \log(\frac{4}{\gamma_2}) \rceil - 1$, $J_4 = \lceil \log_3(\frac{4}{\gamma_2}) \rceil - 1$, we obtain $|\ln\,(1 + \frac{1}{3}) - S_{J_2}(\frac{1}{3})| \leq \frac{\gamma_2}{4}$ and so on. Let $N \geq \lceil \log \frac{4L_0}{\gamma_2} \rceil$. Then $N \geq \max(J_1, ..., J_4)$ and defining

$$y := L_0[S_N(\frac{1}{3}) + S_N(\frac{1}{2})] + S_N(\frac{1}{3}) + S_N(y_1),$$

we have $|\ln(x) - y| \leq \gamma_2$. The total time needed for the computation of y is $O(L_0 + N)$.

(iii) Let $x \geq 1$ (the proof for $x < 1$ is almost the same). Starting with the interval $[1, x]$ and iterating interval halving we need at most $\lceil \log(\frac{x}{\gamma_3}) \rceil$ iterations to find a y with $|y - \sqrt{x}| \leq \gamma_3$. Hence with $N \geq \lceil \log \frac{x}{\gamma_3} \rceil$ the total time needed is $O(N)$ and since the encoding length of x is L, y has encoding length $O(L + N)$.

□

Lemma 2.8 *Let $a_1, \ldots, a_n, b, \gamma$ be rational numbers with encoding length at most L, $b \geq 1$ and $0 < \gamma \leq 1$. Let $\delta > 0$ and let $P_1, \ldots, P_n, Q$ be polynomials in $n, m, \frac{1}{\delta}$ with $P_i, Q \geq 1$, $|a_i| \leq P_i$ and $|b| \leq Q$ for all $i = 1, \ldots, n$. Let $P = \sum_{i=1}^{n} P_i$ and denote by P_i, P, Q also the numbers $P_i(n, m, \frac{1}{\delta}), P(n, m, \frac{1}{\delta})$ and $Q(n, m, \frac{1}{\delta})$.*

(i) Let T_N be the N-th degree Taylor polynomial of the exponential function with

$$N = 10\lceil P \rceil \lceil \log Q \rceil + n + \lceil \log \frac{n+1}{\gamma} \rceil.$$

Then a rational number c approximating $\ln b$ and the numbers $T_N(a_i c)$ can be computed in $O(\max(n, P \log Q) + \log \frac{1}{\gamma})$-time such that the inequality

$$|\prod_{i=1}^{n} e^{a_i \ln b} - \prod_{i=1}^{n} T_N(a_i c)| \leq \gamma$$

holds uniformly for all $a_1, \ldots, a_n$ as above.

(ii) Let T_N be the N-th degree Taylor polynomial of the exponential function with $N = 10\lceil P \rceil + n + \lceil \log \frac{n+1}{\gamma} \rceil$. Then each $T_N(a_i)$ can be computed in $O(\max(n, P) + \log \frac{1}{\gamma})$-time such that the inequality

$$|\prod_{i=1}^{n} e^{a_i} - \prod_{i=1}^{n} T_N(a_i)| \leq \gamma$$

holds uniformly for all $a_1, \ldots, a_n$ as above.

(iii) The encoding length of $T_N(a_i c)$ (resp. of $T_N(a_i)$) is $O(L[\max(n, P \log Q) + \log(\frac{1}{\gamma})]^2)$ (resp. $O(L[\max(n, P) + \log(\frac{1}{\gamma})]^2)$.

□

Proof. (i) and the firstpart of (iii): To shorten notation set $\eta = \frac{\gamma}{n+1} 2^{-n} e^{-2P\lceil \log Q \rceil}$, $\xi = \frac{\gamma}{n+1} e^{-3P\lceil \log Q \rceil}$, $L_0 = \lfloor \log Q \rfloor$, $N_1 = \lceil \log \frac{4(n+1)L_0}{\gamma} \rceil + 3\lceil P \rceil \lceil \log Q \rceil$ and observe that $N_1 \geq \lceil \log \frac{4L_0}{\xi} \rceil \geq \lceil \log \frac{4\lfloor \log b \rfloor}{\xi} \rceil$, Using Lemma 2.7 (ii) we can compute a rational number $c \geq 0$ such that

$$|\ln b - c| \leq \xi \tag{2}$$

in time

$$O(L_0 + N_1) = O(\max(\log n, P \log Q) + \log(\frac{1}{\gamma})), \tag{3}$$

and the encoding length of c is $O(LN_1) = O(L[\max(\log n, P \log Q) + \frac{1}{\gamma}])$. By the mean value theorem, there is a $\nu \in [c, \ln b]$ (or $\nu \in [\ln b, c]$, if $\ln b \leq c$) with

$$\left| e^{\sum_{i=1}^{n} a_i \ln b} - e^{\sum_{i=1}^{n} a_i c} \right| = |\ln b - c| |\sum_{i=1}^{n} a_i| e^{\nu \sum_{i=1}^{n} a_i}$$

$$
\begin{aligned}
&\le \xi P e^{P(1+\log Q)} \quad \textit{with } 2 \\
&\le \xi P e^{2P\lceil \log Q \rceil} \\
&\le \xi e^{3P\lceil \log Q \rceil} \\
&\le \frac{\gamma}{n+1}.
\end{aligned} \tag{4}
$$

Now we approximate $e^{\sum_{i=1}^{n} a_i c}$: put $N = 10\lceil P\rceil\lceil \log Q\rceil + n + \lceil \log \frac{n+1}{\gamma}\rceil$ and let T_N be the N-th degree Taylor polynomial of the exponential function. Since $N \ge 8\lceil |a_i c_i|\rceil + \lceil \log \frac{1}{\eta}\rceil$, we can invoke Lemma 2.7 (i): having precomputed c as above, $T_N(a_i c)$ can be computed in time

$$
O(N) = O(\max(n, P\log Q) + \log \frac{1}{\gamma}), \tag{5}
$$

its encoding length is

$$
O(LN_1N) = O(L[\max(n, P\log Q) + \log(\frac{1}{\gamma})]^2)
$$

and for each $i = 1, \ldots, n$ the estimate

$$
|e^{a_i c} - T_N(a_i c)| \le \eta \tag{6}
$$

holds. Furthermore, because $|\ln b - c| \le \xi \le 1$

$$
\begin{aligned}
|T_N(a_i c)| &\le 1 + e^{a_i c} \\
&\le 1 + e^{a_i(1+\ln b)} \\
&\le 2e^{2P_i\lceil \log Q\rceil}.
\end{aligned}
$$

So, for any product $\prod_{i=1}^{n} F_i$ where F_i is either $e^{a_i c}$ or $T_N(a_i c)$ we have

$$
\prod_{i=1}^{n} F_i \le 2^n e^{2\sum_{i=1}^{n} P_i\lceil \log Q\rceil} = 2^n e^{2P\lceil \log Q\rceil} \tag{7}
$$

Employing the triangle inequality n-times and using (4), (6), (7) we get

$$
\begin{aligned}
\left|\prod_{i=1}^{n} e^{a_i \ln b} - \prod_{i=1}^{n} T_N(a_i c)\right| &\le n2^n e^{2P\lceil \log Q\rceil}\eta \\
&= \frac{n\gamma}{n+1} \\
&\le \gamma.
\end{aligned}
$$

By (3) and (5) the total computation time of each $T_N(a_i c)$ is

$$
O(N) = O(\max(n, P\log Q) + \log \frac{1}{\gamma}).
$$

(ii) Apply the proof of (i) skipping the computation of the logarithms.

□

The next lemma will be needed to show the monotonicity of the pessimistic estimator. Its proof is an easy exercise.

Lemma 2.9 *Let $f_1, \ldots, f_n$ be a finite and monotone decreasing sequence of real numbers. Let $\mu > 0$ and let $g_1, \ldots, g_n$ be a sequence with $|f_l - g_l| \le \mu$. The sequence $h_1, \ldots, h_n$ defined by $h_l = g_l + 2(2n - l)\mu$ for each $l = 1, \ldots, n$ is monotone decreasing.*

□

2.3 $0-1$ Random Variables

Let $m \in \mathbb{N}$. We define m large deviation events as follows:

We are given n mutually independent $0-1$ random variables $X_1, \ldots, X_n$ defined through $Prob(X_j = 1) = \tilde{x}_j$ and $Prob(X_j = 0) = 1 - \tilde{x}_j$ for some rational numbers $0 \leq \tilde{x}_j \leq 1$. For $1 \leq i \leq m$, $1 \leq j \leq n$ let w_{ij} denote rational weights with $0 \leq w_{ij} \leq 1$ and denote by ψ_i the random variables

$$\psi_i = \sum_{j=1}^{n} w_{ij} X_j.$$

For $1 \leq i \leq m$ let $\lambda_i > 0$ be rational numbers and define the event $E_i^{(+)}$ by

$$\text{“}\psi_i \leq \mathbb{E}(\psi_i) + \lambda_i\text{”}$$

and let $E_i^{(-)}$ denote the event

$$\text{“}\psi_i \geq \mathbb{E}(\psi_i) - \lambda_i\text{”}.$$

Furthermore set $E = \bigcap_{i=1}^{m} E_i$ where E_i is either $E_i^{(+)}$ or $E_i^{(-)}$. For each event E_i let $f(\lambda_i)$ be the upper bound on $\mathbb{P}(E_i^c)$ given by the corresponding large deviation inequality in Theorem 2.1 or 2.2, so $f(\lambda_i) = \exp(-\frac{2\lambda_i^2}{n})$ or $f(\lambda_i) = \exp(-\frac{\beta_i^2 \mathbb{E}(\psi)}{d})$ with $d = 2, 3$. Suppose that for some $0 < \epsilon < 1$ the strict inequality

$$\sum_{i=1}^{m} f(\lambda_i) < 1 - \epsilon \tag{8}$$

is satisfied. Then Theorem 2.1 resp. 2.2 imply $\mathbb{P}(\bigcap_{i=1}^{m} E_i) \geq \epsilon$, hence $\bigcap_{i=1}^{m} E_i$ is not empty and we wish to find a vector $x \in \bigcap_{i=1}^{m} E_i$ in deterministic time bounded by a polynomial in n, m and $\log \frac{1}{\epsilon}$.

Before we start with the proof, we briefly sketch the main steps. We wish to construct pessimistic estimators for the events E_i^c. For example, let E_i be the event “$\psi_i \leq \mathbb{E}(\psi_i) + \lambda_i$”. Conditioning on $(X_1, \ldots, X_l) = (y_1, \ldots, y_l)$ with $y_j = 0, 1$ and $1 \leq l \leq n$, Markoff's inequality and the independence of the X_j's imply

$$\begin{aligned} \mathbb{P}[E_i^c | y_1, \ldots, y_l] &\leq e^{-\lambda_i t_i} \mathbb{E}(e^{t_i \psi_i} | y_1, \ldots, y_l) \\ &= e^{-\lambda_i t_i} \prod_{j=1}^{n} \mathbb{E}(e^{t_i w_{ij} X_j} | y_1, \ldots, y_l). \end{aligned}$$

In the most complicated case t_i is of the form $t_i = \ln s_i$ and we have to approximate the factors

$$\mathbb{E}(e^{w_{ij} X_j \ln s_i} | y_1, \ldots, y_l).$$

This can be done by Taylor polynomials and such polynomials will define a pessimistic estimator. The crucial point is that the accurancy of approximation or in other words the degree of such polynomials must be chosen carefully in order to guarantee both, a fast polynomial running time of the approximation procedure and the pessimistic estimator properties.

First let us consider the Angluin-Valiant bound. Before we continue, we put a soft technical restriction on the deviation terms λ_i.

Deviation parameter in the Angluin-Valiant bound:

Let $\lambda_i = \beta_i \mathbb{E}(\psi_i)$. If E_i is an event of the form $E_i^{(-)}$, then E_i is non trivial only, if $\lambda_i < \mathbb{E}(\psi_i)$, which — assuming $\mathbb{E}(\psi_i) > 0$ — is equivalent to $\beta_i < 1$. But in the proof of Theorem 2.2 (b) (see [18], proof of corollary 5.2 (b)) an optimal choice of the parameter t_i introduced in (1) requires that t_i is a real function in $\mathbb{E}(\psi_i)$ and β_i and has a singularity at $\beta_i = 1$. For this reason we assume that

$$\beta_i \leq 1 - \frac{1}{n^{\kappa_1}} \tag{9}$$

for some $\kappa_1 > 0$. Note that the restriction above is only a technical assumption and does not affect the applicability of derandomization to the integer programming examples considered in this paper.

Theorem 2.10 *(Algorithmic Angluin-Valiant Inequality) Let $0 < \epsilon < 1$ and $E_1, \ldots, E_m$ be a collection of events estimated by the Angluin-Valiant bound. Suppose that (8) and (9) are satisfied. Then $\mathbb{P}(\bigcap_{i=1}^m E_i) \geq \delta$ and a vector $x \in \bigcap_{i=1}^m E_i$ can be constructed in $O\left(mn^2 \log \frac{mn}{\epsilon}\right)$ time.*

Proof. In the following we will concentrate on the proof of the required running time. Space consideration can be done in parallel passing through the proof and repeatedly using Lemma 2.8 (ii). Since this requires only tedious calculations, but in principle should be clear, we omit the details.

Case 1: $m = 2$

Set $\lambda_i = \beta_i \mathbb{E}(\psi_i)$. Let E_1 be the event:

$$\psi_1 \leq \mathbb{E}(\psi_1) + \lambda_1$$

and let E_2 be the event

$$\psi_2 \geq \mathbb{E}(\psi_2) - \lambda_2.$$

All other combination of events can be treated in the same way. The basic functions V_1, V_2 from which we will derive the pessimistic estimator are defined as follows. For $1 \leq l \leq n$ let $y_1, \ldots, y_l$ be chosen from $\{0, 1\}$. The upper bounds for the conditional probabilities are

$$\mathbb{P}[E_i^c | y_1, \ldots, y_l] \leq e^{-t_i \lambda_i} \mathbb{E}(e^{\sigma_i t_i \psi_i}),$$

where $\sigma_1 = +1$, $\sigma_2 = -1$ and an optimal choice of t_i gives the Angluin-Valiant bound. According to McDiarmids proof of the Angluin-Valiant inequality [18] $t_i = \ln s_i$ with

$$s_1 = \frac{(\mathbb{E}(\psi_1) + \lambda_1)(n - \mathbb{E}(\psi_1))}{\mathbb{E}(\psi_1)(n - \mathbb{E}(\psi_1) - \lambda_1)}, \tag{10}$$

$$s_2 = \frac{\mathbb{E}(\psi_2)(n - \mathbb{E}(\psi_2) + \lambda_2)}{(n - \mathbb{E}(\psi_2))(\mathbb{E}(\psi_2) - \lambda_2)}, \tag{11}$$

The event " $\psi_1 \leq \mathbb{E}(\psi_1) + \lambda_1$ " with $\lambda_2 = \beta_1 \mathbb{E}(\psi_1)$:

Let s_1 be as in (10) and define for $l \geq 1$

$$V_l^{(1)}(y_1, \ldots, y_l) = e^{-(\mathbb{E}(\psi_1) + \lambda_1) \ln s_1} e^{\sum_{j=1}^l w_{1j} y_j \ln s_1} \prod_{j=l+1}^n [\tilde{x}_j e^{w_{1j} \ln s_1} + 1 - \tilde{x}_j]$$

and for $l = 0$

$$V_0^{(1)} = e^{-(\mathbb{E}(\psi_1) + \lambda_1) \ln s_1} \prod_{j=1}^n [\tilde{x}_j e^{w_{1j} \ln s_1} + 1 - \tilde{x}_j].$$

The event " $\psi_2 \geq \mathbb{E}(\psi_2) - \lambda_2$ " with $\lambda_2 = \beta_2 \mathbb{E}(\psi_1)$:

With s_2 as in 11 define for $l \geq 1$

$$V_l^{(2)}(y_1,\ldots,y_1) = e^{-(\lambda_2 - \mathbb{E}(\psi_2))\ln s_2} e^{-\sum_{j=1}^{l} w_{2j} y_j \ln s_2} \prod_{j=l+1}^{n} [\tilde{x}_j e^{-w_{2j}\ln s_2} + 1 - \tilde{x}_j]$$

and for $l = 0$

$$V_0^{(2)} = e^{-(\lambda_2 - \mathbb{E}(\psi_2))\ln s_2} \prod_{j=1}^{n} [\tilde{x}_j e^{w_{2j}\ln s_2} + 1 - \tilde{x}_j].$$

To unify the notation put $w_{i0} \equiv 0$ $(i = 1,2)$. Then the $V_l^{(i)}$'s $(i = 1,2)$ can be rewritten as

$$V_l^{(i)}(y_1,\ldots,y_l) = \prod_{j=0}^{n} \mathbb{E}(e^{a_{ij}\ln s_i}),$$

with

$$a_{ij} = \begin{cases} -((-1)^{i-1}\mathbb{E}(\psi_i) + \lambda_i) & : \ j = 0 \\ (-1)^{i-1} w_{ij} y_j & : \ j = 1,\ldots,l \\ (-1)^{i-1} w_{ij} X_j & : \ j = l+1,\ldots,n \end{cases}$$

Note that X_j is our random variable, so for $j \geq l+2$ the a_{ij}'s are random variables, too. By McDiarmid's proof of the Angluin-Valiant inequality ([18], proof of corollary 5.2 (b)) we have

$$\mathbb{P}(E_i^c | y_1,\ldots,y_l) \leq V_l^{(i)}(y_1,\ldots,y_l) \tag{12}$$

and using the assumption (8)

$$\mathbb{P}(E_1^c) + \mathbb{P}(E_2^c) \leq V_0^{(1)} + V_0^{(2)} \leq e^{-\frac{\beta_1^2 E(\psi_1)}{3}} + e^{-\frac{\beta_2^2 E(\psi_2)}{2}} < 1 - \epsilon. \tag{13}$$

In view of condition (a) and (c) of Definition 2.6 the functions V_l are the right upper bounds from which the pessimistic estimator should be derived. We will apply Lemma 2.8 First we show that the s_i's are polynomially bounded.

Claim: Let $\kappa = max(1,\kappa_1)$. Then $s_i \leq 4n^\kappa$ for $i = 1,2$.

Proof of the Claim: In order to bound s_i from above we introduce $n+1$ dummy random variables $X_{n+1},\ldots,X_{2n+1}$ and multiply each such X_j with weight 0. This changes neither the expectation nor the bounds nor the proof of Theorem 2.2 except that we have to consider $2n+1$ instead of n. Since $\mathbb{E}(\psi_1) \leq n$ we have

$$s_1 = \frac{(\mathbb{E}(\psi_1) + \lambda_1)(2n+1-\mathbb{E}(\psi_1))}{\mathbb{E}(\psi_1)(2n+1-\mathbb{E}(\psi_1)-\lambda_1)} \quad \leq \quad 2(2n+1).$$

Furthermore with the assumption (9) and using $\mathbb{E}(\psi_2) \leq n$

$$\begin{aligned} s_2 = \frac{\mathbb{E}(\psi_2)(2n+1-\mathbb{E}(\psi_2)+\lambda_2)}{(2n+1-\mathbb{E}(\psi_2))(\mathbb{E}(\psi_2)-\lambda_2)} \quad &\leq \quad \frac{2n+1-(1-\beta_2)\mathbb{E}(\psi_2)}{(2n+1-\mathbb{E}(\psi_2))(1-\beta_2)} \\ &\leq \quad \frac{2n+1}{(n+1)n^{-\kappa_2}} \\ &\leq \quad 2n^{\kappa_1}. \end{aligned}$$

We invoke Lemma 2.8 (i) : Set $\gamma = \frac{\epsilon}{2(4n-1)}$ and $Q = 2n^{3\kappa}$. Since $|a_{i0}| \leq 2n$ for $i = 1,2$ and $|a_{ij}| \leq 1$ for $j = 1,\ldots,l+1$, we can set for each $i = 1,2$, $P_0 = 2n$ and $P_j = 1$ for $j = 1,\ldots,l+1$, hence $P = \sum_{j=0}^{l} P_j \leq 3n$. With N as in in Lemma 2.7 we have

$$N = 10\lceil P\rceil\lceil\log Q\rceil + n + \lceil\log\frac{n+1}{\gamma}\rceil = O(n\log n + \log\frac{1}{\epsilon}). \tag{14}$$

Let T be the $N-$th degree Taylor polynomial of the exponential function. Then Lemma 2.8 (i) implies that for each $i = 1, 2$ the estimate

$$\left| \prod_{j=0}^{n} e^{a_{ij} \ln s_i} - \prod_{j=0}^{n} T(a_{ij} c_i) \right| \leq \gamma \tag{15}$$

uniformly holds for all a_{ij} depending on $y_1, \ldots, y_l$ and for every i the rational rational numbers c_i and $T(a_{ij} c_i)$ can be computetd in $O(n \log n + \log \frac{1}{\epsilon})$ time. Note that this estimation is uniform for all a_{ij}, because

$$\sum_{j=1}^{l} |a_{ij}| \leq \sum_{i=0}^{l} P_i = P \leq 3n.$$

Taking expectation and using the independence of the X_j and (15) we conclude for each $i = 1, 2$

$$|V_l^{(i)}(y_1, \ldots, y_l) - \prod_{j=0}^{n} \mathbb{E}(T(a_{ij} c_i))| \leq \gamma. \tag{16}$$

We proceed to the definition of the pessimistic estimator. For $i = 1, 2$ define

$$T_i(y_1, \ldots, y_l) = \prod_{j=0}^{n} \mathbb{E}(T(a_{ij} c_i)),$$

and

$$T(y_1, \ldots, y_l) = (T_1 + T_2)(y_1, \ldots, y_l).$$

Let U_l be a sequence of functions defined by

$$U_l(y_1, \ldots, y_l) = T(y_1, \ldots, y_l) + 4(2n - l)\gamma. \tag{17}$$

Furthermore let $U_l^{\min}(x_1, \ldots, x_l)$ be iteratively defined by the following procedure.

j=1: Let x_1 be the value from $\{0, 1\}$, which minimizes the function $y \to U_1(y)$. Set

$$U_1^{\min}(x_1) := U_1(x_1).$$

j=l: Suppose that $x_1, \ldots, x_{l-1}$ have been chosen from $\{0, 1\}$ and $U_{l-1}^{\min}(x_1, \ldots, x_{l-1})$ has been defined. Let x_l be the minimizer of $y \to U_l(x_1, \ldots, x_{l-1}, y)$, $y \in \{0, 1\}$, and define

$$U_l^{\min}(x_1, \ldots, x_{l-1}, x_l) := U_l(x_1, \ldots, x_{l-1}, x_l).$$

Let $(U_l^{\min})$ denote the sequence $U_1^{\min}(x_1), \ldots, U_n^{\min}(x_1, \ldots, x_n)$.
First we show that the sequence $(U_l^{\min})$ satisfies the conditions (a), (b) and (c) of Definition 2.6. Define

$$V_l = V_l^{(1)} + V_l^{(2)}. \tag{18}$$

Then by (16) the inequality

$$|T(y_1, \ldots, y_l) - V_l(y_1, \ldots, y_l)| \leq 2\gamma \tag{19}$$

holds *uniformly* for all $y_1, \ldots, y_l \in \{0, 1\}$.

Condition (a):

By (13), (19) and (17)

$$\begin{aligned}\mathbb{P}(E_1^c \cup E_2^c | x_1, \ldots, x_l) &\leq (V_l^{(1)} + V_l^{(2)})(x_1, \ldots, x_l) \\ &\leq (T_1 + T_2)(x_1, \ldots, x_l) + 2\gamma \\ &\leq U_l(x_1, \ldots, x_l) + 4(2n - l)\gamma.\end{aligned}$$

But by definition, $U_l(x_1, \ldots, x_l) + 4(2n - l)\gamma = U_l^{\min}(x_1, \ldots, x_l)$.

Condition (b):
In order to apply Lemma 2.9 put

$$f_l = \min[V_l(y_1, \ldots, y_{l-1}, 1), V_l(y_1, \ldots, y_{l-1}, 0)]$$

and

$$g_l = \min[T(y_1, \ldots, y_{l-1}, 1), T(y_1, \ldots, y_{l-1}, 0)].$$

Using (19) we have

$$|f_l - g_l| \leq 2\gamma$$

for all $l = 1, \ldots, n$. Since $f_1, \ldots, f_n$ is monotonely decreasing, Lemma 2.9 implies that the sequence (U_l^{min}) possesses the same property.

Condition (c):
With condition (b), using $\min(V_1(1), V_1(0)) \leq V_0$ and (13) we get

$$\begin{aligned}U_1^{min}(x_1) &= T_1(x_1) + T_2(x_1) + 4(2n - 1)\gamma \\ &\leq \min(V_1(1), V_1(0)) + 2\gamma + 4(2n - 1)\gamma \\ &\leq V_0 + 2\gamma + 4(2n - 1)\gamma \\ &= V_0^{(1)} + V_0^{(2)} + 2\gamma + 4(2n - 1)\gamma \\ &< 1 - \epsilon + 2\gamma + 4(2n - 1)\gamma \\ &= 1.\end{aligned}$$

We are done, if we can show an overall running time of $O(mn^2 \log \frac{mn}{\epsilon})$. Let us fix $1 \leq l \leq n$ and consider the Taylor approximation for $V_l^{(1)}$. The argumentation for $V_l^{(2)}$ goes similar. First note that

$$V_l^{(1)}(y_1, \ldots, y_l) = V_{l-1}^{(1)}(y_1, \ldots, y_{l-1}) \frac{1}{\mathbb{E}(e^{a_{1l} \ln s_1})} e^{w_{1l} y_{1l} \ln s_1}. \tag{20}$$

According to Lemma 2.7 (i) and with N as in (14) we can compute c_1, $\mathbb{E}(T(a_{1l}c_1))$ and $\mathbb{E}(T(w_{1l}y_{1l}c_1))$ in $O(N) = O(n \log n + \log \frac{1}{\epsilon})$-time. In the first step the approximation of

$$e^{-(\mathbb{E}(\psi_1) + \lambda_1) \ln s_1} \prod_{j=1}^{n} [\tilde{x}_j e^{w_{1j} \ln s_1} + 1 - \tilde{x}_j]$$

requires the computation of $n+1$ Taylor polynomials. This takes $O(n[n \log n + \log \frac{1}{\epsilon}])$ time. Then by induction and using the recursion (20) the total time for the computaion of $U_l(x_1, \ldots, x_l)$ is

$$O\left(n\left[n \log n + \log \frac{1}{\epsilon}\right] + \sum_{i=2}^{n}\left(n \log n + \log \frac{1}{\epsilon}\right)\right) = O\left(n^2 \log \frac{n}{\epsilon}\right).$$

Case 2 $m \geq 2$:

Note that for arbitrary m the same proof goes through, if we replace ϵ by $\frac{2\epsilon}{m}$ and define

$$U_l(y_1,\dots,y_l) = (T_1+\dots+T_m)(y_1,\dots,y_l) + 2m(2n-l)\gamma.$$

Then we get a worst case running time of

$$O(mn[n\log n + \log\frac{m}{\epsilon}]) = O(mn^2\log\frac{mn}{\epsilon})$$

and the theorem is proved.

□

The algorithmic version of the Chernoff-Hoeffding-Bernstein bound can be derived similarily.

Theorem 2.11 *(Algorithmic Chernoff-Hoeffding Inequality) Let $0 < \epsilon < 1$ and $E_1,\dots,E_m$ be events estimated by the Chernoff-Hoeffding inequality. Suppose that (8) is satisfied. Then*

$$\mathbb{P}(\bigcap_{i=1}^m E_i) \geq \epsilon$$

and a vector $x \in \bigcap_{i=1}^m E_i$ can be constructed in $O\left(mn[n+\log\frac{m}{\epsilon}]\right)$-time.

Proof. We follow the argumentation in the proof of the algorithmic Angluin-Valiant inequality. Let the events E_i be as there. The Chernoff-Hoeffding bound is

$$f(\lambda_i) \leq \exp(-\frac{2\lambda_i^2}{n}).$$

According to the proof of the Chernoff-Hoeffding inequality (Theorem 2.1) as given in [18] the parameters t_i are $t_i = \frac{4\lambda_i}{n}$. Therefore we do not have to compute logarithms and can spare a log-factor. Because trivially $\lambda_i \leq n$, we have $O(t_i\lambda_i + nt_i) = O(n)$, thus the exponent of

$$e^{-t_i\lambda_i}\mathbb{E}(e^{t_i\psi_i})$$

is $O(n)$. So due to Lemma 2.7 (ii) the degree of the approximating Taylor polynomial as well as the time to evaluate such a polynomial is only $O(n+\log(\frac{nm}{\epsilon})) = O(n+\log(\frac{m}{\epsilon}))$. The rest of the proof can be carried out as in Theorem 2.10.

□

2.4 The Case $\Omega = \prod_{j=1}^n\{1-\tilde{x}_j, -\tilde{x}_j\}$

In this subsection we consider the Alon-Spencer bounds. We can argue as in the section above, with minor modifications of the notation. We are given n mutually independent random variables defined through $Prob(X_j = 1-\tilde{x}_j) = \tilde{x}_j$ and $Prob(X_j = -\tilde{x}_j) = 1-\tilde{x}_j$ for some rational numbers $0 \leq \tilde{x}_j \leq 1$. For $1 \leq i \leq m$, $1 \leq j \leq n$ let w_{ij} be rational weights from $\{0,1\}$ and denote by ψ_i the random variables

$$\psi_i = \sum_{j=1}^n w_{ij}X_j.$$

Put $p_i = \mathbb{E}(\psi_i)/n_i$ where $n_i = \sum_{j=1}^n w_{ij}$ and let $\lambda_i > 0$ be rational numbers. For $1 \leq i \leq m$ let $E_i^{(+)}$ be the event "$\psi_i \leq +\lambda_i$" and let $E_i^{(-)}$ denote the event "$\psi_i \geq -\lambda_i$". Furthermore set $E = \bigcap_{i=1}^m E_i$ where E_i is either $E_i^{(+)}$ or $E_i^{(-)}$. For each event E_i let $f(\lambda_i)$ be the upper bound for $\mathbb{P}(E_i^c)$ as given by the corresponding large deviation inequalities in Theorem 2.3 or 2.4, so

$f(\lambda_i) = \exp(-\frac{2\lambda^2}{n})$ or $f(\lambda_i) = \exp(-\frac{\lambda_i^2}{2p_in_i} + \frac{\lambda_i^3}{2(p_in_i)^2})$ or $f(\lambda_i) = \exp(-\frac{\lambda_i^2}{2p_in_i})$. Suppose that for some $0 < \epsilon < 1$

$$\sum_{i=1}^{m} f(\lambda_i) < 1 - \epsilon. \tag{21}$$

Furthermore, we need again some technical assumption to avoid singularities of parameters used in the proof of the underlying bounds.

Deviation Parameter in the Alon-Spencer Bound:
We need to consider Theorem 2.4 (a) only If $\sum_{j=1}^{n} w_{ij} > 0$, then we assume that

$$\sum_{j=1}^{n} w_{ij}\tilde{x}_j \geq \frac{1}{n^{\kappa_2}} \tag{22}$$

for some $\kappa_2 \geq 1$ and

$$\lambda_1 = O(n^{\kappa_3}) \tag{23}$$

for some $\kappa_3 \geq 1$. The derandomization result is:

Theorem 2.12 *Let $0 < \epsilon < 1$ and $E_1, \ldots, E_m$ be events satisfying (22), (23) and (21). Then $I\!P(\bigcap_{i=1}^{m} E_i) \geq \epsilon$ and a vector $x \in \bigcap_{i=1}^{m} E_i$ can be constructed in $O\left(mn^2 \log \frac{mn}{\epsilon}\right)$-time.*

Proof: In view of proof of Theorem 2.10 it is sufficient to consider the case $m = 2$.

Let $1 \leq l \leq n$ and $y_1, \ldots, y_l$ with $y_i \in \{1 - \tilde{x}_i, -\tilde{x}_i\}$.
The basic functions V_1, V_2 here are:

The event "$\psi_1 \leq \lambda_1$":

Let $t_1 > 0$ and define for $l \geq 1$

$$V_l^{(1)}(y_1, \ldots, y_l) = e^{-t_1\lambda_1} e^{\sum_{j=1}^{l} w_{1j}y_jt_1} \prod_{j=l+1}^{n} [\tilde{x}_j e^{w_{1j}(1-\tilde{x}_j)t_1} + (1-\tilde{x}_j)e^{-w_{1j}\tilde{x}_jt_1}]$$

The event "$\psi \geq -\lambda_2$":

Let $t_2 > 0$ and define for $l \geq 1$

$$V_l^{(2)}(y_1, \ldots, y_l) = e^{-t_2\lambda_2} e^{-\sum_{j=1}^{l} w_{2j}y_jt_2} \prod_{j=l+1}^{n} [\tilde{x}_j e^{-w_{2j}(1-\tilde{x}_j)t_2} + (1-\tilde{x}_j)e^{w_{1j}\tilde{x}_jt_1}].$$

With the following minor modifications the proof can be carried out as in the 0-1 case. The parameters t_i can be choosen according to the proof of Corollary A.7 (respectively the proof of Corollary A.10/Theorem A.13 in [1]): In case of Theorem 2.3, $t_i = \frac{4\lambda_i}{n}$ for $i = 1, 2$ and in the proof of Theorem 2.4 (b), $t_2 = \frac{\lambda_2}{\sum_{j=1}^{n} w_{2j}\tilde{x}_j}$. Therefore the exponents above are rational numbers and in view of Lemma 2.8 we don't have to compute logarithms. In case of Theorem 2.4 (a) $t_1 = \ln(1 + \frac{\lambda_1}{\sum_{j=1}^{n} w_{1j}\tilde{x}_j})$ and by restriction (22), $\frac{\lambda_1}{\sum_{j=1}^{n} w_{1j}\tilde{x}_j} \leq \lambda_1 n^{\kappa_2}$. This will give us according to Lemma 2.8, taking $Q = 1 + \lambda_1 n^{\kappa_2}$ and with γ as in the proof of Theorem 2.10 a running time of $O(\kappa_2 n^2 \log \frac{n\lambda_1}{\epsilon})$. With $\lambda_1 = O(n^{\kappa_3})$ as in restriction (23) and assuming that the κ's are constant, we are done.

$\square$

2.5 Multivalued Random Variables

Finally, we consider multivalued random variables, especially n mutually independent dice. We investigate a situation in which the random variables under consideration have Biomial distribution and thus may apply the tools developed so far. Let n, N be non-negative integers. We are given n mutually independent random variables X_j with values in $\{0, \ldots, N\}$ and probability distribution $Prob(X_j = k) = \tilde{x}_{jk}$ for all $j = 1, \ldots, n$, $k = 1, \ldots, N$ and $\sum_{k=1}^{N} \tilde{x}_{jk} = 1$. Suppose that the $\tilde{x}_{jk}$ are rational numbers with $0 \leq \tilde{x}_{jk} \leq 1$. Let X_{jk} denote the random variable which is 1, if $X_j = k$ and is 0 else. The probability space is

$$\Omega = \{(y_1, \ldots, y_n) \in \prod_{j=1}^{n} \{0,1\}^N ; y_j \in \{0,1\}^N \ , \ \sum_{k=1}^{N} y_{jk} = 1\}.$$

For $1 \leq k \leq N$, $1 \leq i \leq m$, $1 \leq j \leq n$ let $w_{ij}^{(k)}$ be rational weights with $0 \leq w_{ij}^{(k)} \leq 1$. For $i = 1, \ldots, m$ and $k = 1, \ldots, N$ define the sums ψ_{ik} by

$$\psi_{ik} = \sum_{j=1}^{n} w_{ij}^{(k)} X_{jk}. \tag{24}$$

Let $\lambda_{ik} > 0$ be rational numbers. Denote by $E_{ik}^{(+)}$ the event

$$\psi_{ik} \leq \mathbb{E}(\psi_{ik}) + \lambda_{ik} \tag{25}$$

and by $E_{ik}^{(-)}$ the event

$$\psi_{ik} \geq \mathbb{E}(\psi_{ik}) - \lambda_{ik} \tag{26}$$

Let (E_{ik}) be a collection of mN such events. We invoke the Angluin-Valiant inequality. As in the $0-1$ case let $f(\lambda_{ik})$ be the upper bounds for $\mathbb{P}(E_{ik}^c)$ given by the inequality under consideration. We suppose that

$$\sum_{i=1}^{m} \sum_{k=1}^{N} f(\lambda_{ik}) < 1 - \epsilon. \tag{27}$$

for some $0 < \epsilon < 1$ and assume that the events satisfy conditions 9.

Theorem 2.13 *Let $0 < \epsilon < 1$ and E_{ik} be as above satisfying (9) and (27). Then*

$$\mathbb{P}(\bigcap_{i=1}^{m} \bigcap_{k=1}^{N} E_{ik}) \geq \epsilon$$

and a vector $x \in \bigcap_{i=1}^{m} \bigcap_{k=1}^{N} E_{ik}$ can be constructed in $O\left(Nmn^2 \log \frac{Nnm}{\epsilon}]\right)$-time.

Proof: For $j = 1, \ldots, n$ let Ω_j be the j-th copy of the set

$$\{\omega \in \{0,1\}^N ; \sum_{k=1}^{N} y_{jk} = 1\}.$$

The only difference to the proof of Theorem 2.10 is that in each step of the conditional probability method we have to choose a *vector* $y \in \Omega_j$ instead of an integer. This can be done as in the proof of Theorem 2.10, but would give us a running time of $O\left(N^2 mn[n + \log \frac{Nmn}{\epsilon}]\right)$ as there are Nm events, n random variables, and - this increases the running time - for each random variable we have N choices. But in our context a simple observation reduces the running time to

$O\left(Nmn[n\log n + \log\frac{Nm}{\epsilon}]\right)$: consider the first step of the computation of the pessimistic estimator. Let $y_1 \in \Omega_1$ be the vector we are going to select in the first step, in other words, we wish to determine the outcome of the first die. Let E_{ik} be arbitrary, but for a moment fix. Then, because ψ_{ik} is a sum of independent Bernoulli trials, ψ_{ik} is either $\sum_{j=2}^{n} w_{ij}^{(k)} X_{jk} + w_{i1}^{(k)}$ or it is $\sum_{j=2}^{n} w_{ij} X_{jk}$. So, for this ψ_{ik} we have to approximate only two upper bounds for the conditional probabilities. Each such bound is the product of $O(n)$ factors of the form $\exp(a_i \ln b_i)$ for some rational numbers a_i, b_i. For each of these factors the approximation time is $O\left(n\log n + \log\frac{Nm}{\epsilon}\right)$, thus for the product we need $O\left(n[n\log n + \log\frac{Nm}{\epsilon}]\right)$ time. (see also the proof of Theorem 2.10. We do this for all events E_{ik} and get a time of $O\left(Nmn[n\log n + \log\frac{Nm}{\epsilon}]\right)$. In the second step, after having selected the first vector from Ω_1, we can use the update argument at the end of the proof of Theorem 2.10 and get a time of $O\left(Nm[n\log n + \log\frac{Nm}{\epsilon}]\right)$. Summing up over all the n steps, we get a overall running time of

$$O\left(Nmn[n\log n + \log\frac{Nm}{\epsilon}]\right) = O\left(Nmn^2\log\frac{Nnm}{\epsilon}]\right).$$

□

3 Integer Programming

3.1 A General Integer Program of Packing Type

Let $\mathbb{Z}_+$ be the set of non-negative integers and let $\mathbb{Q}_+$ be the set of non-negative rational numbers. Let us consider the following integer program:

$$\max\{c^T x \ ; \ Ax \leq b, x \in \mathbb{Z}_+^n\},$$

where $b \in \mathbb{Q}_+^m$, A is a $m \times n$ matrix with rational entries $a_{ij} \in [0,1]^n$ and c is a rational vector $c \in [0,1]^n$.

Let us denote by P the polytope $\{x \in \mathbb{Q}_+^n; Ax \leq b\}$ and by P_I its integer skeleton $P \cap \mathbb{Z}_+^n$. The LP relaxation, where the entries x_j of x can take arbitrary non-negative rational values, can be solved in polynomial-time with standard linear programming algorithms. Let $y \geq 0, y \in \mathbb{Q}_+^n$ be an optimal solution vector found by linear programming. If we try to apply the known $0-1$ randomized rounding method directly, we get problems due to the fact that we are rounding to arbitrary integers and we must guarantee that the rounded vector is in P_I, with positive probability. There are two more or less obvious randomized rounding methods for rounding the entries y_j to an integer, but both have drawbacks:

(a) The perhaps most obvious rounding procedure is to round y_j to $\lceil y_j \rceil$ or to $\lfloor y_j \rfloor$. This can be done in a randomized way performing n independent Bernoulli trials ξ_j, defined through $Prob(\xi_j = 1) = y_j - \lfloor y_j \rfloor$ and $Prob(\xi_j = 0) = 1 - y_j + \lfloor y_j \rfloor$. Let y^I be the rounded vector with entries $\lfloor y_j \rfloor + \xi_j$ and denote by $\lfloor y \rfloor$ the vector with entries $\lfloor y_j \rfloor$. Invoking the Angluin-Valiant inequality we can prove $y^I \in P_I$ as follows. Let $b^{red} \in \mathbb{Q}^m$ be the decreased vector with entries $b_i^{red} := b_i - (A\lfloor y \rfloor)_i$. Then with Theorem 2.2 (a)

$$\begin{aligned} Prob(y^I \notin P_I) &= Prob(\exists\ i(Ay^I)_i > b_i) \\ &= Prob(\exists i\ (A\xi)_i > b_i - (A\lfloor y \rfloor)_i) \\ &= Prob(\exists i\ (A\xi)_i > (1+1)\frac{1}{2} b_i^{red}) \\ &\leq \sum_{i=1}^{m} e^{-\frac{b_i^{red}}{12}} \end{aligned}$$

and we can conclude that $Prob(y^I \in P_I) > 0$, if the last inequality is strictly less than 1. This indeed is the case under the typical assumption of randomized rounding in integer programming, i.e. if $b_i^{red} = \Omega(\ln m)$ for all i (the constant here is 12). See also [23], analysis of k-matching. But even if $b_i = \Omega(\ln m)$, it may happen that the decreased right hand side b^{red} drops below the lower bound for b_i and the analysis fails. This is the reason why the $0-1$ randomized rounding scheme of [23] cannot be applied directly.

(b) An intuitive better idea is to perform a more flexible rounding in which by chance some y_j can become much bigger or smaller than $\lceil y_j \rceil$. One extreme way to do so is to split off each y_j into $2\lfloor y_j \rfloor$ $0-1$ "segments" of value 0.5 and one segment of value $y_j - \lfloor y_j \rfloor$. This complete splitting enforces $b_i^{red} = b_i$ and the $0-1$ randomized rounding scheme is applicable: for each y_j randomly round the values of the segments to 0 or 1 with probabilities equal to the segment values. The j-th entry of the rounded vector y^I then is the sum over all the rounded segments corresponding to y_j. Hence we have reduced the problem to $0-1$ randomized rounding, and since $b_i^{red} = b_i$, we have $Prob(y^I \in P_I) > 0$, provided that $b_i = \Omega(\ln m)$ for all i. Unfortunately, this is *not* a polynomial-time rounding algorithm as the number of random variables depends on the magnitude of numbers appearing in the fractional solution.

Our strategy is to compromize between these two extreme roundings. Let $0 < \epsilon < 1$. The goal is to derive an $(1-\epsilon)$-factor approximation of the integer optimum. It is achieved in 3 steps.

- (Randomized Rounding) First we split off each y_j in a fixed integer part y_j^{fix} and a sufficiently big roundable part y_j^{var} with $y_j = y_j^{fix} + y_j^{var}$ (Algorithm Split(ϵ)). The sizes of the roundable parts y_j^{var} are responsible for the number of random variables we use. In Lemma 3.1 we show that at most $O(\frac{m \log m}{\epsilon})$ $0-1$ random variables are needed to ensure that for all i, $b_i^{red} = \Omega(\frac{\log m}{\epsilon})$, whenever $b_i = \Omega(\frac{\log m}{\epsilon})$. Then for each $j = 1, \ldots, n$ we set $k_j = \lfloor y_j^{var} \rfloor$ and define $2k_j + 1$ independent $0-1$ random variables $\chi_1, \ldots, \chi_{2k_j+1}$. The rounded vector $x \geq 0$, $x \in \mathbb{Z}^n$ will have entries
$$x_j := y_j^{fix} + \sum_{l=1}^{2k_j+1} \chi_l,$$
$j = 1, \ldots, n$ (Algorithm ROUNDING).

- (Analysis) In Theorem 3.2 we show with the Angluin-Valiant inequality (Theorem 2.2) that x satisfies $Ax \leq b$ **and** $c^T x \geq (1-\epsilon) c^t x_{opt}$ with probability at least $\frac{1}{4}$, where x_{opt} is an optimal integer solution.

- (Derandomization) Finally, we will derandomize the algorithm via the algorithmic Angluin-Valiant inequality.

In the whole analysis we need two important parameters, b_ϵ and c_ϵ:

$$b_\epsilon := \lceil \frac{6(2-\epsilon)}{\epsilon^2} \rceil \lceil \log(2m) \rceil \quad and \quad c_\epsilon := \frac{16}{\epsilon^2} \tag{28}$$

Algorithm SPLIT(ϵ)

INPUT: The fractional optimal solution $y = (y_1, \ldots, y_n)$ with $y_j \geq 0$ and $0 < \epsilon < 1$.

OUTPUT: For each y_j an integer $y_j^{fix} \geq 0$ and a rational number $y_j^{var} \geq 0$ with $y_j = y_j^{fix} + y_j^{var}$.

begin
Initialization: **Set** for all $j = 1, \ldots, n$
  $y_j^{fix} := \lfloor y_j \rfloor$,
  $y_j^{var} := y_j - \lfloor y_j \rfloor$,
 for each $i = 1, \ldots, m$ **do**
  While $b_i - (Ay^{fix})_i < b_\epsilon$ **do**
    choose $y_j \in \{y_1^{fix}, \ldots, y_n^{fix}\}$ with $a_{ij} > 0$ and $y_j \geq 1$.

set $y_j^{fix} := y_j^{fix} - 1$ and $y_j^{var} := y_j^{var} + 1$.
end
While $c^T y - c^T y^{fix} < c_\epsilon$ **do**
choose $y_j \in \{y_1^{fix}, \ldots, y_n^{fix}\}$ with $c_j > 0$ and $y_j \geq 1$.
set $y_j^{fix} := y_j^{fix} - 1$ and $y_j^{var} := y_j^{var} + 1$.
end
end

The next lemma follows immediately.

Lemma 3.1 *Let* $b_\epsilon = \lceil \frac{6(2-\epsilon)}{\epsilon^2} \rceil \lceil \log(2m) \rceil$ *and* $c_\epsilon = \frac{16}{\epsilon^2}$ *as in (28). If* $b_i \geq b_\epsilon$ *for all* $i = 1, \ldots, m$ *and* $\sum_{j=1}^{n} c_j y_j \geq c_\epsilon$*, then SPLIT(*$\epsilon$*) generates for each* b_i *at most* $O(\frac{\log m}{\epsilon^2})$ *random variables and computes* y^{fix} *in* $O(\frac{m \log m}{\epsilon^2})$ *time such that*

$$b_i - (Ay^{fix})_i \geq b_\epsilon \text{ and } c^T y - c^T y^{fix} \geq c_\epsilon \tag{29}$$

for all $i = 1, \ldots, m$.

□

Now we can define the randomized rounding procedure. For each $j = 1, \ldots, n$ set $k_j = \lfloor y_j^{var} \rfloor$ and define $2k_j + 1$ independent $0 - 1$ random variables $\chi_1, \ldots, \chi_{2k_j+1}$ by

$$\begin{aligned} Prob\,(\chi_l = 1) &= \frac{1}{2}\left(1 - \frac{\epsilon}{2}\right) \\ Prob(\chi_l = 0) &= 1 - \frac{1}{2}\left(1 - \frac{\epsilon}{2}\right) \\ Prob\,(\chi_{2k_j+1} = 1) &= (y_j - \lfloor y_j \rfloor)\left(1 - \frac{\epsilon}{2}\right) \\ Prob\,(\chi_{2k_j+1} = 0) &= 1 - (y_i - \lfloor y_j \rfloor)\left(1 - \frac{\epsilon}{2}\right), \end{aligned}$$

$1 \leq l \leq 2k_j$.

Algorithm **ROUNDING**

1. For each $l = 1, \ldots, 2k_j + 1$ set independently χ_l to 0 or 1 with probabilities defined as above.
2. Output is the rounded vector $x \geq 0$, $x \in \mathbb{N}^n$ with components

$$x_j := y_j^{fix} + \sum_{l=1}^{2k_j+1} \chi_l,$$

$j = 1, \ldots, n$.

□

Theorem 3.2 *Let* $0 < \epsilon \leq \frac{9}{10}$ *and* $b_\epsilon = \lceil \frac{6(2-\epsilon)}{\epsilon^2} \rceil \lceil \log(2m) \rceil$*. Suppose that* $b_i \geq b_\epsilon$ *for all* $i = 1, \ldots, m$ *and* $c_1 + \ldots + c_{b_\epsilon} \geq \frac{16}{\epsilon^2}$*. Then an integer vector* $x \in \mathbb{Z}^n$*,* $x \geq 0$ *with* $Ax \leq b$ *can be constructed in polynomial-time such that*

$$c^T x \geq (1 - \epsilon) c^T y \geq (1 - \epsilon) c^T x_{opt}.$$

Proof. Note that the somewhat strange restriction $\epsilon \le \frac{9}{10}$ is necessary to satisfy condition (9), but has no influence on the quality of approximation, since we want to approximate a maximum. We divide the proof into 3 steps. First we show that the vector x is in P_I with probability at least $\frac{1}{2}$. Then it will be proved that with probability at least $\frac{3}{4}$, $c^T x$ is an $(1-\epsilon)$-approximation of $c^T x_{opt}$. Hence with probability at least $\frac{1}{4}$ both is true and in the third and last step we derandomize using the algorithmic Angluin-Valiant inequality.

Claim 1: $\mathbb{P}(Ax \le b) \ge \frac{1}{2}$.

Proof. Let b_i^{red} be the reduced right hand side with

$$b_i^{red} := b_i - (Ay^{fix})_i$$

For each $j = 1, \ldots, n$ let ξ_j be the random variable

$$\xi_j := \sum_{l=1}^{2k_j+1} \chi_l$$

and let $\xi \in \mathbb{Z}_+^n$ denote the vector with entries ξ_j. For $i = 1, \ldots, m$ define Ψ_i by

$$\Psi_i := (A\xi)_i.$$

Then

$$\begin{aligned}
\mathbb{E}(\Psi_i) &= (Ay^{var})_i(1 - \frac{\epsilon}{2}) \\
&= \sum_{j=1}^n a_{ij}(y_j - y_j^{fix})(1 - \frac{\epsilon}{2}) \\
&\le (b_i - \sum_{j=1}^n a_{ij} y_j^{fix})(1 - \frac{\epsilon}{2}) \\
&= (1 - \frac{\epsilon}{2})(b_i - (Ay^{fix})_i) \\
&= (1 - \frac{\epsilon}{2}) b_i^{red}.
\end{aligned}$$

Taking $\beta_i = \frac{\epsilon}{2-\epsilon}$ for all i we get by the Angluin-Valiant inequality (Theorem 2.2 (a))

$$\begin{aligned}
\mathbb{P}(\Psi_i > b_i^{red}) &= \mathbb{P}(\Psi_i > (1+\beta)(1 - \frac{\epsilon}{2}) b_i^{red}) \\
&\le \exp\left(- \frac{\beta_i^2 \left(1 - \frac{\epsilon}{2}\right) b_i^{red}}{3} \right) \\
&\le \exp(-\log 2m) \\
&= \frac{1}{2m}.
\end{aligned} \tag{30}$$

Hence for all $i = 1, \ldots, m$

$$\begin{aligned}
(Ax)_i &= [Ay^{fix} + \Psi]_i \\
&= (Ay^{fix})_i + \Psi_i \\
&\le (Ay^{fix})_i + b_i^{red} \\
&= b_i
\end{aligned}$$

with probability at least $\frac{1}{2}$.

Claim 2: $\mathbb{P}(c^T x \geq (1-\epsilon)c^T y) \geq \frac{3}{4}$.

Proof. Define the reduced objective function value by

$$z^{red} := c^T y - c^T y^{fix}.$$

Then the random variable $z := c^T \xi$ satisfies $\mathbb{E}(z) = c^T y^{var}(1 - \frac{\epsilon}{2})$.

The vector with 1 in the first b_ϵ entries and 0 elsewhere is feasible, because on the one hand $b_i \geq b_\epsilon$ and on the other hand $c_1 + \ldots + c_{b_\epsilon} \geq \frac{16}{\epsilon^2}$, hence $c^T y \geq \frac{16}{\epsilon^2}$. According to Lemma 3.1 we have $z^{red} \geq \frac{16}{\epsilon^2}$ and setting $\beta_0 = \sqrt{\frac{8}{(2-\epsilon)z^{red}}}$ it is easily verified that

$$(1-\beta)z^{red}(1 - \frac{\epsilon}{2}) \geq (1-\epsilon)z^{red}.$$

Hence by the Angluin-Valiant inequality

$$\mathbb{P}(z < (1-\epsilon)z^{red}) \leq \mathbb{P}(z < (1-\beta)(1-\frac{\epsilon}{2})z^{red}) \leq \frac{1}{4} \tag{31}$$

and Claim 2 is proved. Combining Claim 1 and 2 we conclude that the assertion of the theorem holds at least with probability $\frac{1}{4}$. In order to derandomize this result, we apply the algorithmic Angluin-Valiant inequality (Theorem 2.10). The total number of random variables after the execution of the algorithm SPLIT(ϵ) is $N = n + N_1$ with $N_1 = O(\frac{m \log m}{\epsilon^2})$. Recall that for $i = 1, \ldots, m$, $\beta_0 = \sqrt{\frac{8}{(2-\epsilon)z^{red}}}$ and $\beta_i = \frac{\epsilon}{2-\epsilon}$. Let E_i be the event "$\Psi_i \leq b_i^{red}$", which can be written as "$\Psi_i \leq (1-\beta_i)(1-\frac{\epsilon}{2})b_i^{red}$" and let E_0 be the event "$c^T\xi \geq (1-\beta_0)(1-\frac{\epsilon}{2})z^{red}$". (30) and (31) imply

$$\mathbb{P}(E_0^c) + \sum_{i=1}^{m} \mathbb{P}(E_i^c) \leq e^{-\frac{\beta_0^2(1-\frac{\epsilon}{2})z^{red}}{2}} + \sum_{i=1}^{m} e^{-\frac{\beta_i^2(1-\frac{\epsilon}{2})b^{red}}{3}} \leq \frac{3}{4},$$

and condition (8) is satisfied with constant probability strictly less than 1. In order to apply Theorem 2.10 we must also ensure that the restriction (9) is satisfied which

$$\beta_0 \leq 1 - \frac{1}{N^{\kappa_1}}$$

for some $\kappa_1 > 0$. Using $z^{red} \geq \frac{16}{\epsilon^2}$, $\epsilon \leq \frac{9}{10}$ and assuming $N \geq 2$ (which always is true) we get

$$\beta_0 \leq 1 - \frac{1}{4} \leq 1 - \frac{1}{N^2}.$$

$\square$

In case of all $c_j = 1$, we trivially have $c_1 + \ldots + c_{b_\epsilon} = b_\epsilon$. Furthermore, if A is a $0-1$ matrix, then the corresponding linear program can be solved in strongly polynomial time by the LP algorithm of Tardos [30] and we have

Corollary 3.3 *Let* $0 < \epsilon \leq \frac{9}{10}$ *and* $b_\epsilon = \lceil \frac{6(2-\epsilon)}{\epsilon^2} \rceil \lceil \log(2m) \rceil$. *Suppose that* $c_j = 1$ *for all* $j = 1, \ldots, n$, *A is a $0-1$ matrix and* $b_i \geq b_\epsilon$ *for all* $i = 1, \ldots, m$. *Then an integer vector* $x \in \mathbb{Z}^n, x \geq 0$ *with* $Ax \leq b$ *can be constructed in strongly polynomial time such that*

$$c^T x \geq (1-\epsilon)c^T y \geq (1-\epsilon)c^T x_{opt}.$$

3.2 Resource Constrained Scheduling

An instance of the resource constrained scheduling problem with start times consists of ([13], p. 239):

• A set $\mathcal{J} = \{J_1, \ldots, J_n\}$ of independent jobs. Each job J_j needs a time of one time unit for its completion and cannot be scheduled before its start time r_j, $r_j \in \{1, \ldots, n\}$.

• A set $\mathcal{P} = \{P_1, \ldots, P_m\}$ of identical processors. Each job needs one processor.

• A set $\mathcal{R} = \{R_1, \ldots, R_s\}$ of renewable, but limited resources. This means that at any time all resources are available, but the available amount of each resource R_i is bounded by $b_i \in \mathbb{N}$. For $1 \leq i \leq s$, $1 \leq j \leq n$ let $R_i(j) \in [0, 1]$ be rational resource requirements, indicating that every job J_j needs $R_i(j)$ amount of resource R_i in order to be processed.

The combinatorial optimization problem is:

• Find a schedule (or assignment) $\sigma : \mathcal{J} \mapsto \mathbb{N}$ of *minimal* time length subject to the starting times, processor and resource constraints.

Since the processor requirements can be decribed by introducing an additional resource R_{s+1} with upper bound $b_{s+1} = m$ and defining $R_{s+1}(j) = 1$, the resource constraints are briefly formalized as

$$\forall z \in \mathbb{N}, i \in \{1, \ldots, s+1\} : \sum_{\{j : \sigma(j) = z\}} R_i(j) \leq b_i,$$

where $\{j : \sigma(j) = z\}$ is the set of jobs scheduled at time z. The problem is NP-hard in the strong sense, even if $r_j = 0$ for all $j = 1, \ldots, n$, $s = 1$ and $m = 3$.

According to the standard notation of scheduling problems the unweighted (i.e. $R_i(j) = 0, 1$) version of our problem can be formalized as $P|\text{res} \cdot\cdot 1, r_j, p_j = 1|C_{\max}$. This notation means that the number of identical processors is part of the input (— $P|$ —) that resources are envolved (— res —) that the number of resources and the amount of every resource are part of the input, too (— res $\cdot\cdot$ —), that every job needs at most 1 unit of a resource (— res $\cdot\cdot 1$ —), that start-up times are envolved (— r_j —) and that the processing time of all jobs is equal (— $p_j = 1$ —) and that optimization problem is to finish the last scheduled job as soon as possible (— $|C_{max}$ —). Note that we consider the rational weighted version with $R_i(j) \in \mathbb{Q} \cap [0, 1]$.

The best known approximation algorithm for the problem class $P|\text{res} \cdots, r_j = 0, p_j = 1|C_{\max}$, where the jobs can be processed at any time ($r_j = 0$) and the maximal resource-usage of a job is part of the input, is due to Röck and Schmidt [24]. They showed, employing the polynomial-time solvability of the simpler problem $P2|\text{res} \cdots, r_j = 0, p_j = 1|C_{\max}$, where only 2 processors are given, a $\lceil \frac{m}{2} \rceil$-factor approximation algorithm. Note that Röck and Schmidt's approach is based on the assumption that no starting-times are given, i.e. $r_j = 0$ for all jobs $J_j \in \mathcal{J}$. In fact, their algorithm cannot be used, when starting times are given, since the problem $P2|\text{res} \cdot\cdot 1, r_j, p_j = 1|C_{\max}$ is also NP-complete, so their basis solution cannot be constructed in polynomial-time.

Furthermore, for zero start times Garey et al. constructed with the First-Fit-Decreasing heuristic a schedule of length C_{FFD} which *asymptotically* is a $(s + \frac{1}{3})$-factor approximation, i.e. there is an non negative integer N such that for all $C_{opt} \geq N$

$$C_{FFD} \leq C_{opt}(s + \frac{1}{3}).$$

de la Vega and Lueker [32] improved this result presenting for every $\epsilon > 0$ a linear time algorithm with asymptotic approximation performance $d + \epsilon$.

Given arbitrary start times in $\{1, \ldots, n\}$ we will show the first polynomial-time 2-factor approximation. Let C_{opt} be the integer minimum of our scheduling problem and let the integer C denote the size of the minimal schedule, if we consider the LP relaxation, where fractional assignments of the tasks to scheduling times are allowed. We briefly call solutions to the LP relaxation "fractional schedules" and solutions to the original integer problem "integral schedules". This should not cause any confusion: C is always an integer, only the assignments corresponding to C are fractional.

Theorem 3.4 *For the problem $P|res \cdot\cdot 1, r_j, p_j = 1|C_{\max}$ with rational resource requirements, i.e. $R_i(j) \in \mathbb{Q} \cap [0,1]$ a schedule of size at most $2C_{opt}$ can be found in deterministic polynomial time, provided that $b_i \geq 6\lceil \log(4C(s+1)) \rceil$ for all $i = 1, \ldots, s+1$.*

Remark

• Note that C is at most the sum of n and the maximal start-time, hence the factor $\log(Cs)$ is within the size of the problem input.

• Our results are related to the results of Lenstra, Shmoys and Tardos [17], who gave a 2-factor approximation algorithm for the problem of scheduling independent jobs with different processing times on *unrelated* processors. Their algorithm is essentially a combinatorial rounding procedure rounding the solution of the associated LP. Moreover, they showed that there is no ρ-approximation algorithm for $\rho < 1.5$, unless $P = NP$. Unfortunately their rounding procedure does not apply to the case, when arbitrary resource constraints are given. The reason is that given arbitrary resource constraints, the LP might loose essential combinatorial structures, for example the polyhedron is not pointed anymore (see [17]). This is a typical situation where randomization might be helpful. The significance of the 2-factor approximation is emphasized by the most probable intractability of the problem of finding approximations better than 1.5 in polynomial-time.

Theorem 3.5 *Even if $b_i \in \Omega(\log(Cs))$ for all constraint bounds b_i, there is no polynomial-time ρ-approximation algorithm for $P|res \cdot\cdot 1, r_j, p_j = 1|C_{\max}$ for any $\rho < 1.5$, unless $P = NP$*

Before going into details, we give an outline of the proof of Theorem 3.4. First we must generate a fractional solution, then we have to define randomized rounding. While the first problem is easily solved by standard methods solving at most $\log T$ linear programs, where $T = n + r_{max}$ and r_{max} is the maximal starting-time, in order to find the minimal fractional completion time C, the second problem is non trivial: for each job J_j let x_{jz} be the $0-1$ variable indicating whether or not the job J_j is processed at time z. Then, because we wish to process the job J_j, we must require $\sum_{z=1}^{C} x_{jz} = 1$. Suppose that we have found the fractional completion time C corresponding to $\tilde{x}_{jz}$, $0 \leq \tilde{x}_{jz} \leq 1$ (the fractional optimal assignments of the jobs to the scheduling times) with $\sum_{z=1}^{C} \tilde{x}_{jz} = 1$. A possible and suggestive randomized rounding procedure would be to cast for each job J_j independently a C-faced die with face probabilities $\tilde{x}_{jz}$, where the z-th face represents the choice of the time z for job J_j for all $z = 1, \ldots, C$ and $j = 1, \ldots, n$. Unfortunately, since we have a packing problem it may happen that simple dice casting produces a schedule in which too many jobs are scheduled at the same time requiring more resources than available.

To avoid such problems we enlarge the time interval $\{1, \ldots, C\}$ to $\{1, \ldots, 2C\}$ and consider for each job j a die with $2C$ faces, where for each $z \in \{1, \ldots, C\}$ the faces z and $z + C$ occur with probability $\frac{\tilde{x}_{jz}}{2}$. In this fashion we will generate a schedule within $2C$ and at each time the expected amount of resource R_i will be only $\frac{b_i}{2}$..

Proof of Theorem 3.4:

Let $r_{max} := \max_{j=1,\ldots,n} r_j$ and $T = r_{max} + n$. Then obviously

$$C \leq C_{opt} \leq T \leq 2n.$$

C can be found as follows: Start with an overall deadline $\tilde{C} \in \{1, \ldots, T\}$ and according to [17] check, whether the LP

$$\begin{array}{rcll} \sum_j R_i(j)x_{jz} & \leq & b_i & \forall R_i \in \mathcal{R}, \\ & & & z \in \{1,\dots,T\} \\ \sum_z x_{jz} & = & 1 & \forall J_j \in \mathcal{J} \\ x_{jz} & = & 0 & \forall z < r_j, j \in \{1,\dots,n\} \\ x_{jz} & = & 0 & \forall J_j \in \mathcal{J}, z > \tilde{C} \\ x_{jz} & \in & [0,1]. & \end{array}$$

has a solution. Using binary search it is clear that we will find C having solved at most $\log T$ such LPs. Let $X_1, \dots, X_n$ be mutually independent random variables taking values in $\{1, \dots, 2C\}$, where for each $z \in \{1, \dots, C\}$

$$\mathbb{P}(X_j = z) = \mathbb{P}(X_j = z + C) = \frac{\tilde{x}_{jz}}{2}.$$

For $z \in \{1, \dots, 2C\}$ and $j = 1, \dots, n$ let X_{jz} be the $0-1$ random variable, which is 1, if $X_j = z$ and zero else. For $i = 1, \dots, s+1$ let E_{iz} be the event that at a time $z \in \{1, \dots, 2C\}$ the i-th resource constraint b_i is not violated:

$$\text{“}\sum_{j=1}^{n} R_i(j)X_{jz} \leq b_i\text{”}$$

Obviously

$$\mathbb{E}(\sum_{j=1}^{n} R_i(j)X_{jz}) = \sum_{j=1}^{n} R_i(j)\frac{\tilde{x}_{jz}}{2} \leq \frac{b_i}{2}$$

for all i and z. By the Angluin-Valiant inequality (Theorem 2.2 (a)) and using the assumption $b_i \geq 6\lceil \log(4C(s+1)) \rceil$ for all $i = 1, \dots, s+1$ we have

$$\begin{aligned} \mathbb{P}[E_{iz}^c] &= \mathbb{P}[\sum_{j=1}^{n} R_i(j)X_{jz} > b_i] \\ &= \mathbb{P}[\sum_{j=1}^{n} R_i(j)X_{jz} > (1+1)\frac{1}{2}b_i] \\ &\leq \exp(-\frac{b_i}{6}) \\ &\leq \frac{1}{4C(s+1)}. \end{aligned}$$

We only have events of the form $E^{(+)}$ and according to Remark 2.13 we don't have to care about any restriction for the deviation parameters, and Theorem 2.13 concludes the proof.

□

The negative result is:

Theorem 3.6 *Even if all starting times are zero and $b_i \in \Omega(\log(ns))$ for all $i = 1, \dots, s+1$, it is NP-complete to determine, whether or not the scheduling problem $P|res \cdot \cdot 1, r_j = 0, p_j = 1|C_{\max}$ has a solution with $C_{opt} = 2$.*

Remark Note that Theorem 3.6 implies Theorem 3.5, since it is a special case of problems considered in Theorem 3.4: We have zero starting-times, hence $T = n$ and $C \leq n$. Therefore the NP-completeness of problems with $b_i \in \Omega(\log(ns))$ implies the completeness of problems with $b_i \in \Omega(\log(Cs))$. And finally, an approximation better than a factor $\frac{3}{2}$ would contradict Theorem 3.6: W.l.o.g. assume that $C_{opt} > 1$. If a ρ-approximation algorithm with $\rho < \frac{3}{2}$ outputs 2, then

$C_{opt} = 2$, and if its output is greater or equal 3, then $C_{opt} \geq 3$ (because $\rho < \frac{3}{2}$). Hence we would be able to decide in polynomial-time whether or not $C_{opt} = 2$.

Proof of Theorem 3.5

We give a reduction to the problem of decomposing a graph into two perfect matchings, which is known to be NP-complete [13]. Let $G = (V, E)$ be a graph with $|V| = n'$. For a moment let $K \geq 0$ be an arbitrary integer. We will define a scheduling problem associated to G with n jobs, m processors and s constraints. First we define an auxilliary graph $H = (V(H), E(H))$:
For each node in G introduce K red copies and $2K$ blue copies and let $V(H)$ be the set of these red and blue nodes. Whenever $\{v, w\} \in E$, put an edge between the corresponding red copies $\{v_i, w_i\}$ of v and w for $1 \leq i \leq K$. Let us call all the red copies corresponding to the same node in G a red set. We identify each node of H with a job, so $n = 3n'K$. Considering $m = 3n'K$ identical processors, we get rid of the processor constraints.

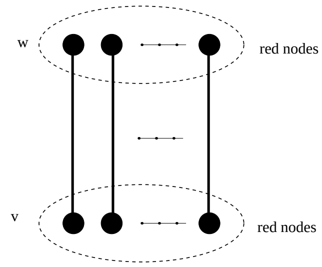

Figure 1: The Graph H

Let us define three type of resource constraints A, B and C corresponding to subsets of $V(G)$ and $V(H)$ as follows:

Type A:

For each set of three nodes (u, v, w) of G with at least two induced edges define a resource $R_{(u,v,w)}$ with upper bound $2K$ and suppose that any job associated to a red copy of one of this three nodes (u, v, w) needs one unit of $R_{(u,v,w)}$ in order to be processed.

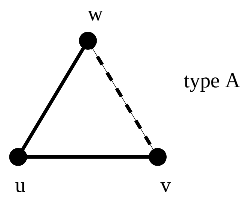

Figure 2: Type A resources

Type B: Whenever v is a node of G with degree two or more, define a resource R_v with upper bound $K(deg(v) - 1)$. Suppose that any job associated with a red copy of one of the neighbours of v needs one unit of R_v in order to be processed.

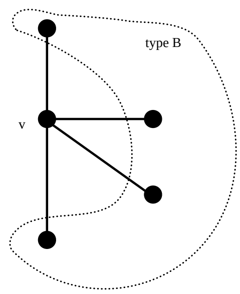

Figure 3: Type B resources

Type C: Define for every red node v_i and every set S_K with K of its corresponding blue nodes a resource $R(v_i, S_K)$ with bound K, and suppose that each job in $S_K \cup \{v_i\}$ needs one unit of $R(v_i, S_K)$.

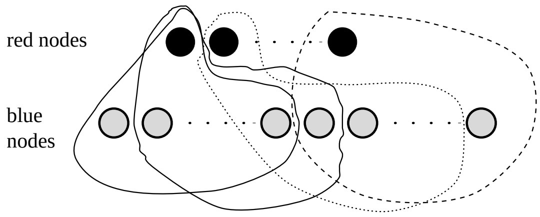

Figure 4: Type C resources

Due to the resource constraint of type C the *key observation* is that in a feasible schedule of length 2 all the red copies of the *same* node $u \in G$ must be scheduled at the *same time*. This can be seen as follows. Let us assume for a moment that this is not true. Then there is a $v \in V$ with at least one red copy v' scheduled at the time 1 and at least one red copy v'' scheduled at time 2. We can schedule, due to resource constraint of type C with bound K, at most $K-1$ blue copies of v at time 1 and therefore must schedule the remaining blue copies of v at time 2, violating some resource constraints of type C.

Hence the problem is whether or not the red jobs can be scheduled in two times *without* splitting off the red sets.

We show: *There is a partitioning of G into 2 perfect matchings if and only if there is a feasible schedule of size 2.*

(a) If there is a feasible schedule, put the nodes of G corresponding to *red* nodes (or jobs) being scheduled at time 1 in a set V_1, and the remaining nodes of G in a set V_2. Since a feasible schedule does not split off the red sets, V_1 and V_2 build a partition of the nodes of G. They induce 2 perfect matchings: Every resource constraint of type B ensures that at least one neighbour of a node $v \in V_i$ is in the same set as v itself, while the constraints of type A ensure that for every node $v \in V_i$ at most one neighbour is in the same set as v itself. Hence the induced degree of v is one and we have constructed two perfect matchings.

(b) Let V_1, V_2 be a partitioning of G into two perfect matchings. (If there are isolated nodes in G, then there are no perfect matchings. Pairs of nodes with degree 1 we put into V_1). Schedule the red copies of nodes in V_1 at time 1 and its blue copies at time 2. Schedule the red copies of nodes in V_2 at time 2 and its blue copies at time 1. Using the matching property it is easily verified that this is a feasible schedule.

The proof is complete, if we can show the logarithmic growth of the constraint bounds m and b_i for $i = 1, \ldots, r$. For this we must specify K. Taking $K = \log n'$ it is easily verified that the number of constraints s is

$$s = O((n')^3 + n' + \binom{2\log n'}{\log n'} \log n') = O((n')^c)$$

for some constant c. Since all our constraint bounds are $\Omega(K)$, we finally can show by a straight forward computation that $K \geq \alpha \log(ns)$ for some constant $\alpha \geq 0$.

□

Remark Theorem 3.5 says that there is no polynomial-time approximation algorithm within a factor $\rho < 1.5$i, unless $P = NP$. This should not be misinterpreted. Theorem 3.6 makes clear that the pathological instances here are instances whose optimal schedule is 2. But it might be possible that for instances with larger optimal schedules better approximation factors can be achieved. Indeed, meanwhile we could prove this and gave in [29] a comprehensive discussion of the complexity of resource constrained scheduling.

4 Conclusion

(a) The running time of the algorithmic Chernoff-Hoeffding inequalities is $O(mn^2 \log \frac{mn}{\epsilon})$, while the basic conditional probability methods runs in $O(mn)$-time. It is an interesting problem to close this gap as much as possible.

(b) In our applications to integer programming we had to assume that the constraint vector $b = (b_1, \ldots, b_m)$ posseses components in $\Omega(\log m)$. It remains an open problem, if approximation algorithms can be given, even if $b_i = O(\log m)$.

(c) For resource constrained scheduling we showed a polynomial-time 2-factor approximation algorithm and also that there does not exist a substantially better approximation algorithms, unless $P = NP$. Extension of this result has been given in [29].

References

[1] N. Alon, J. Spencer, P. Erdős; *The probabilistic method.* John Wiley & Sons, Inc. 1992.

[2] N. Alon; A parallel algorithmic version of the Local Lemma. *Random Structures and Algorithms, Vol.2, No.4, (1991), 367-378.*

[3] D. Angluin, L.G. Valiant: Fast probabilistic algorithms for Hamiltonion circuits and matchings. *J. Comp. Sys. Sci., Vol. 18, (1979), 155–193.*

[4] K. Azuma, Weighted sums of certain dependent variables. *Tohoku Math. Journ. 3, (1967), 357-367.*

[5] B. Berger, J. Rompel; Simulating $(log^c n)$-wise independence in NC. *JACM, 38 (4), (1991), 1026 – 1046.*

[6] J. Błazewicz, K. Ecker, G. Schmidt, J. Węglarz; *Scheduling in computer and maufacturing systems.* Springer-Verlag, Berlin (1993).

[7] R. P. Brent; Fast multiple-precision evaluation of elementary functions. *JACM, Vol. 23,(1976), 242 - 251.*

[8] A. Feldstein, P.R. Turner; Overflow, underflow, and severe loos of significance in floating point addition and subtraction. *IMA J. Numer. Analysis Vol. 6, (1986), 241 - 251.*

[9] H. Chernoff; A measure of asymptotic efficiency for test of a hypothesis based on the sum of observation. *Ann. Math. Stat. 23, (1952), 493-509.*

[10] V. Chvatal, The tail of the hypergeometric distribution. *Disc. Math. 25, (1979), 285-287.*

[11] P. Erdös, J.L. Selfridge; On a combinatorial game. J. Comb.Th., Ser.A 14, 298-301, (1973).

[12] M. R. Garey, R. L. Graham, D. S. Johnson, A.C.-C. Yao; Resource constrained scheduling as generalized bin packing. *J. Comb.Th.Ser.A, 21 (1976), 257 - 298.*

[13] M. R. Garey, D. S. Johnson; *Computers and Intractability.* W. H. Freeman and Company, New York (1979).

[14] M. Grötschel, L. Lovász, A. Schrijver; *Geometric algorithms and combinatorial optimixation.* Springer-Verlag (1988).

[15] W. Hoeffding; On the distribution of the number of success in independent trials. *Ann. Math. Stat. 27, (1956), 713-721.*

[16] K. L. Krause, V. Y. Shen, H .D. Schwetmann; Analysis of several job-scheduling algorithms for a model of multiprogramming computer systems. *JACM 22 (1975) 522–550. Erratum: JACM 24, (1977), p. 527.*

[17] J. K. Lenstra, D. B. Shmoys, E. Tardos; Approximating algorithms for scheduling unrelated parallel machines. *Math. Programming, 46, (1990), 259 – 271.*

[18] C. McDiarmid; On the method of bounded differences. in:*Surveys in Combinatorics, 1989. J. Siemons, Ed.: London Math. Soc. Lectures Notes, Series 141, Cambridge University Press, Cambridge, England 1989.*

[19] K. Mehlhorn; *Data structures and algorithms 1: Sorting and Searching.* Sringer-Verlag (1984).

[20] R. Motwani, J. Naor, M. Naor; The probabilistic method yields deterministic parallel algorithms. *Proceedings 30the IEEE Conference on Foandation of Computer Science (FOCS'89), (1989), 8 – 13.*

[21] J. Naor, M. Naor; Small bias probability spaces: Efficient constructions and applications. *SIAM J. Computing, 22 (4), 1993, 838 – 856.*

[22] P. Raghavan, C. D. Thompson; Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica 7 (4), (1987), 365-374.*

[23] P. Raghavan; Probabilistic construction of deterministic algorithms: approximating packing integer programs. *J. Comp. Sys. Sci., 37, (1988), 130-143.*

[24] H. Röck, G. Schmidt; Machine aggregation heuristics in shop scheduling. *Math. Oper. Res. 45(1983) 303–314.*

[25] J. P. Schmidt, A. Siegel, A. Srinivasan; Chernoff-Hoeffding Bounds for Applications with Limited Independence. In: *Proceedings of the fourth ACM-SIAM Symposium on Discrete Algorithms (1993), 331–340*

[26] J. Spencer; *Ten lectures on the probabilistic method.* SIAM, Philadelphia (1987).

[27] A. Srivastav, P. Stangier; Weighted fractional and integral k-matching in hypergraphs. *Disc.Appl.Math. 57, (1995), 225 – 269.*

[28] A. Srivastav, P. Stangier; Integer multicommodity flows with reduced demands. in: *T. Lengauer (eds.), Proceedings of the First European Symposium on Algorithms (ESA'93), Bonn/Bad Honnef (1993), pp. 360 - 372, Lecture Notes in Computer Science, Vol. 726, Springer-Verlag.*

[29] A. Srivastav, P. Stangier; Tight aproximations of resource constrained scheduling problems. in: *J. van Leuuwen (ed), Proceedings of the Second Annual European Symposium on Algorithms (ESA'94), Utrecht (1994), pp. 307 - 318, Lecture Notes in Computer, Vol. 855, Springer Verlag.*

[30] E. Tardos; A strongly polynomial algorithm to solve combinatorial linear programs. *Oper. Res., Vol.34, (1986), 250 - 256.*

[31] C-K. Yap; Towards exact geometric computation. in: *Proceedings of the 5th Canadian Conference on Computational Geometrie, University of Waterloo, (1993), pp. 405 - 419.*

[32] W. F. de la Vega, C. S. Luecker; Bin packing can be solved in within $1 - \epsilon$ in linear time.*Combinatorica, 1 (1981), 349 - 355.*