

**CS6999**

**Probabilistic Methods in Integer Programming**

---

**Randomized Rounding**

**Andrew D. Smith**

**April 2003**

## Background

- â Integer Programming
- â The Probabilistic Method
- â Randomization and Approximation
- â The Covering Radius of a Code

## Randomized Rounding

- â The Algorithm
- â Probabilistic Analysis

## Handling Feasibility

- â Covering and Packing Problems
- â Scaling Variables

## Derandomization

- â Conditional Probabilities
- â Pessimistic Estimators

## Advanced Techniques

- â Positive Correlation
- â Limited Dependence

maximizing or minimizing a linear function over a polyhedron

given    matrix  $A \in \mathbb{R}^{m \times n}$   
          vector  $b \in \mathbb{R}^m$   
          vector  $c \in \mathbb{R}^n$

find a vector  $x$  optimizing

$$\max\{cx \mid Ax \leq b\}$$

$$\min\{cx \mid x \geq 0; Ax \geq b\}$$

Alexander Schrijver (1986)

“Theory of linear and integer programming”

maximizing or minimizing a linear function over a polyhedron

given    matrix  $A \in \mathbb{R}^{m \times n}$   
         vector  $b \in \mathbb{R}^m$   
         vector  $c \in \mathbb{R}^n$

find a vector  $x$  optimizing

$$\max\{cx \mid Ax \leq b\}$$

$$\min\{cx \mid x \geq 0; Ax \geq b\}$$

$x$  is the solution vector

$\max cx$  or  $\min cx$  are objective functions

$Ax \leq b$ ,  $Ax \geq b$ ,  $x \geq 0$  are constraints on the solution

Alexander Schrijver (1986)

“Theory of linear and integer programming”

maximizing or minimizing a linear function over a polyhedron

given    matrix  $A \in \mathbb{R}^{m \times n}$   
          vector  $b \in \mathbb{R}^m$   
          vector  $c \in \mathbb{R}^n$

find a vector  $x$  optimizing

$$\max\{cx \mid Ax \leq b\}$$

$$\min\{cx \mid x \geq 0; Ax \geq b\}$$

$x$  is the solution vector

$\max cx$  or  $\min cx$  are objective functions

$Ax \leq b$ ,  $Ax \geq b$ ,  $x \geq 0$  are constraints on the solution

linear programming	$x \in \mathbb{R}^n$	polynomial time
integer programming	$x \in \mathbb{Z}^n$	NP-complete (decision version)

Alexander Schrijver (1986)  
“Theory of linear and integer programming”

solving optimization problems over discrete structures

famous integer programs

- Facility Location
- Knapsack Problem
- Job Shop Scheduling
- Travelling Salesperson

all NP-hard!

some exact solution techniques

- Branch and Bound
- Dynamic Programming
- Cutting Planes
- Ellipsoid/Separation

exact algorithms fail eventually!

solving optimization problems over discrete structures

famous integer programs

- Facility Location
- Knapsack Problem
- Job Shop Scheduling
- Travelling Salesperson

all NP-hard!

some exact solution techniques

- Branch and Bound
- Dynamic Programming
- Cutting Planes
- Ellipsoid/Separation

exact algorithms fail eventually!

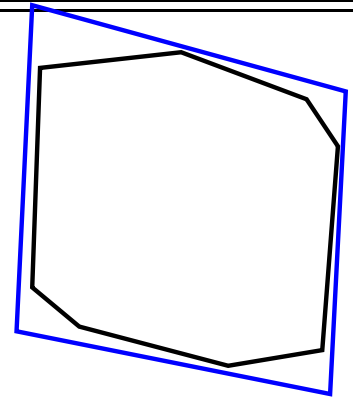
what happens when no exact algorithm is fast enough  
but some solution must be found?

Find an approximately optimal solution

more on this shortly . . .

relaxation: remove constraints

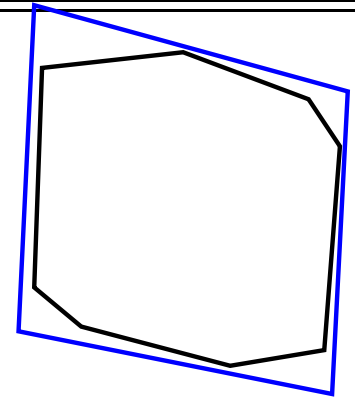
geometrically: expand (and simplify) the polyhedron  
translation: allow more vectors to be solutions





relaxation: remove constraints

geometrically: expand (and simplify) the polyhedron  
translation: allow more vectors to be solutions

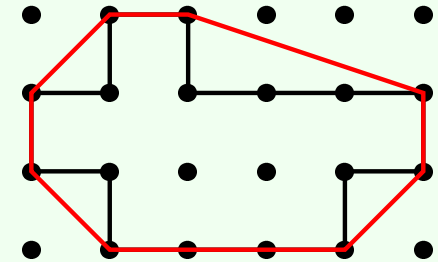


fractional relaxation: remove integrality constraints

very important technique in solving integer programs

$$x \in \mathbb{Z}^n \mapsto \bar{x} \in \mathbb{R}^n$$

- recall that linear programs can be solved in polynomial time (e.g. primal dual interior point)
- fractional relaxations for NP-hard problems can be very easy (e.g. fractional knapsack reduces to sorting)
- provide intuition about (geometric?) structure of problem
- (important concept) integrality gap =  $c\bar{x}/cx$



translation: allow more vectors to be solutions

very important technique in solving integer programs

- true understanding of an NP-complete combinatorial optimization problem requires understand its fractional relaxation

## Randomized Algorithms

any algorithm with behavior depending on random bits (**we assume a source of random bits**)

quality measures

time complexity

number of random bits required

probability of success

Motwani & Raghavan (1995) “Randomized algorithms”

D. Hochbaum (1995) “Approximation algorithms for NP-hard problems”

## Randomized Algorithms

any algorithm with behavior depending on random bits (**we assume a source of random bits**)

quality measures

time complexity

number of random bits required or probability of success

## Approximation Algorithms

polynomial time algorithm for an NP-hard optimization problem

performance ratio

$\frac{\text{optimal}}{\text{approximate}}$  (maximization)

$\frac{\text{approximate}}{\text{optimal}}$  (minimization)

performance ratios are better when closer to 1 and cannot be  $< 1$

quality measures

time complexity

performance ratio

Motwani & Raghavan (1995) “Randomized algorithms”

D. Hochbaum (1995) “Approximation algorithms for NP-hard problems”

## Randomized Algorithms

any algorithm with behavior depending on random bits (**we assume a source of random bits**)

quality measures

time complexity

number of random bits required or probability of success

## Approximation Algorithms

polynomial time algorithm for an NP-hard optimization problem

performance ratio

$\frac{\text{optimal}}{\text{approximate}}$  (maximization)

$\frac{\text{approximate}}{\text{optimal}}$  (minimization)

performance ratios are better when closer to 1 and cannot be  $< 1$

quality measures

time complexity

performance ratio

combine the two [Randomized Approximation Algorithms](#)

Motwani & Raghavan (1995) “Randomized algorithms”

D. Hochbaum (1995) “Approximation algorithms for NP-hard problems”

Trying to prove that a structure with certain desired properties exists, one defines an appropriate probability space of structures and then shows that the desired properties hold in this space with positive probability. — Alon & Spencer

Alon & Spencer (2000) “The probabilistic method” 2nd edition

Trying to prove that a structure with certain desired properties exists, one defines an appropriate probability space of structures and then shows that the desired properties hold in this space with positive probability. — Alon & Spencer

importance for algorithms

- can be used to show a solution exists
- and to show that a solution exists with particular qualities
- and to show that a solution exists where an algorithm will find it
- essential tool in design and analysis of randomized algorithms

Alon & Spencer (2000) “The probabilistic method” 2nd edition

Trying to prove that a structure with certain desired properties exists, one defines an appropriate probability space of structures and then shows that the desired properties hold in this space with positive probability. — Alon & Spencer

importance for algorithms

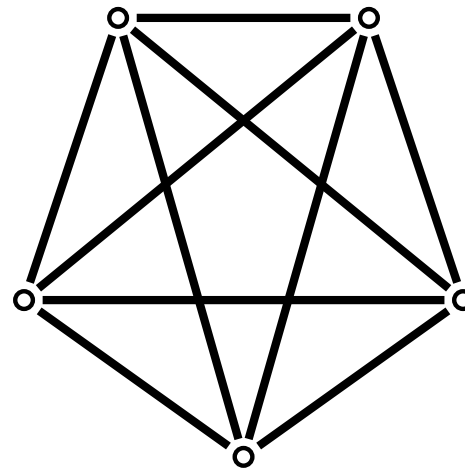
- can be used to show a solution exists
- and to show that a solution exists with particular qualities
- and to show that a solution exists where an algorithm will find it
- essential tool in design and analysis of randomized algorithms

**caveat** non-constructive by definition

Alon & Spencer (2000) “The probabilistic method” 2nd edition

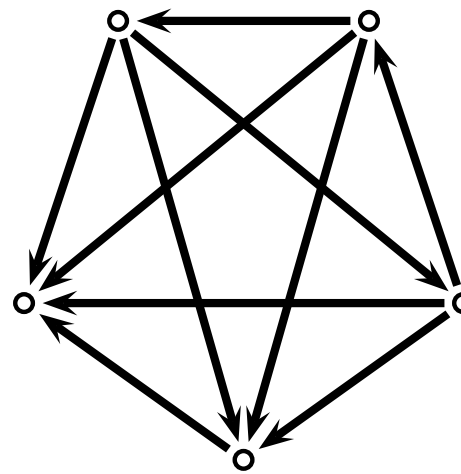


$K_5$  is the complete graph on 5 vertices



$K_5$  is the complete graph on 5 vertices

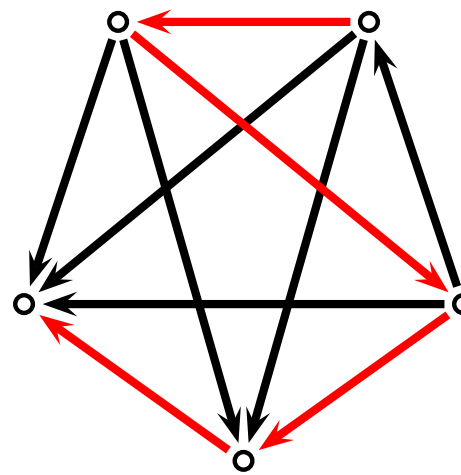
A 5 player tournament is  
an orientation of  $K_5$



$K_5$  is the complete graph on 5 vertices

A 5 player tournament is  
an orientation of  $K_5$

A Hamiltonian path touches  
each vertex exactly once

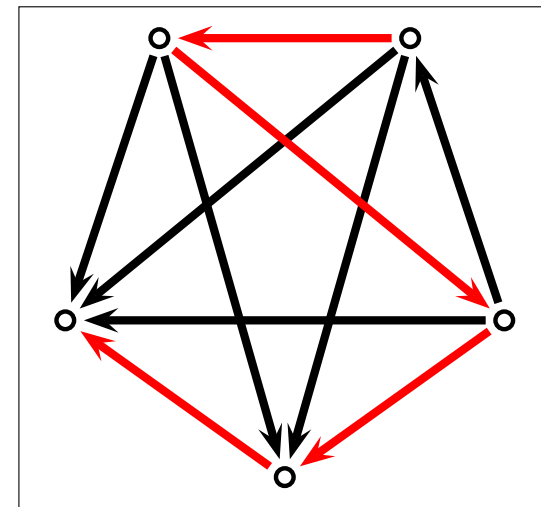


## Theorem

There is a tournament with  $n$  players and at least

$$n!2^{(1-n)}$$

Hamiltonian paths.

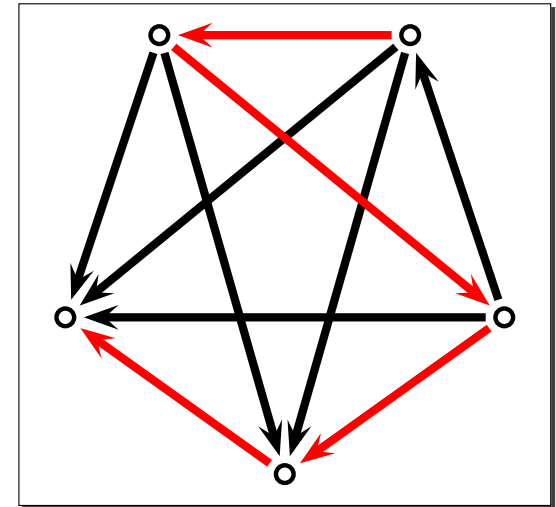


## Theorem

There is a tournament with  $n$  players and at least

$$n!2^{(1-n)}$$

Hamiltonian paths.



## Proof

Let  $T$  be a random tournament on  $n$  players and let  $X$  be the number of Hamiltonian paths in  $T$ .

For each permutation  $\sigma$  let  $X_\sigma$  be the indicator random variable for  $\sigma$  giving a Hamiltonian path - i.e., satisfying  $(\sigma(i), \sigma(i+1)) \in T$  for  $1 \leq i \leq n$ .

Then  $X = \sum X_\sigma$  and

$$E[X] = \sum E[X_\sigma] = n!2^{(1-n)}.$$

Thus some tournament has at least  $E[X]$  Hamiltonian paths.

formal problem statement

instance: a code  $\mathcal{F} \subseteq \{0, 1\}^n$ ,  $\mathcal{F} = \{s_1, \dots, s_m\}$ .

objective: find a vector  $z$  minimizing  $\max_i d_H(s_i, z)$ .

- $d_H$  is the **Hamming distance** (number of mismatching positions)
- the vector  $z$  is called the **center** of  $\mathcal{F}$
- special case of the Closest String problem (highly approximable)
- important problem in **clustering** and **classification**

$$\mathcal{F} = \{s_1, s_2, s_3, s_4\}$$

$$z = 0110111100$$

$$s_1 = 0111111110$$

$$s_2 = 000110110$$

$$s_3 = 1110111100$$

$$s_4 = 1111001111$$

$$d_H(s_4, z) = 6$$

Frances & Litman (1997) "On covering problems of codes"

identify vector  $s_i$  in  $\mathcal{F}$  with row  $a_i$  of  $A$   
 for each position  $j$ , row  $a_i$  has **two positions**

example transformation

$$\begin{array}{l} s_1 = 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1 \ 1 \\ a_1 = 10 \ 01 \ 10 \ 10 \ 01 \ 10 \ 01 \ 01 \ 10 \ 10 \end{array}$$

$$\begin{array}{l} (s_j = 0) \Leftrightarrow (x_{j_1} = 0) \wedge (x_{j_2} = 1) \\ (s_j = 1) \Leftrightarrow (x_{j_1} = 1) \wedge (x_{j_2} = 0) \end{array}$$

$$\begin{array}{ll} \text{minimize} & d \\ \text{subject to} & \sum_{j=1}^n (a_{ij_1} x_{j_1} + a_{ij_2} x_{j_2}) \leq d, \\ & \forall 1 \leq j \leq n, x_{j_1} + x_{j_2} = 1, x_{j_1}, x_{j_2} \in \{0, 1\} \end{array}$$

fractional relaxation

$$\bar{x}_{j_1}, \bar{x}_{j_2} \in [0, 1]$$

$$\begin{array}{ll} \mathcal{F} = \{s_1, s_2, s_3, s_4\} & A = [a_1, a_2, a_3, a_4]^T \\ s_1 = 0111111110 & a_1 = 01101010101010101001 \\ s_2 = 0000110110 & a_2 = 01010101101001101001 \\ s_3 = 1110111100 & a_3 = 10101001101010100101 \\ s_4 = 1111001111 & a_4 = 10101010010110101010 \end{array}$$

$$\begin{array}{ll} a_4 = 10101010010110101010 & \Rightarrow a_4 x = 6 \\ x = 10010110010101011010 & \end{array}$$

input: a fractional vector  $\bar{x} \in \mathbb{R}^n$  (a solution to the linear relaxation)  
 output: a random vector  $x \in \mathbb{Z}^n$  distributed according to  $\bar{x}$

start with the solution to the linear programming relaxation

```

RANDOMIZED ROUNDING( $\bar{x}$ ) {
  for each position  $j$  of  $x$ 
    with probability  $\bar{x}_{j_1}$  set  $x_{j_1} \leftarrow 1$  and  $x_{j_2} \leftarrow 0$ 
    otherwise set  $x_{j_1} \leftarrow 0$  and  $x_{j_2} \leftarrow 1$ 
  return  $x$ 
}
```

all solution  
 vectors produced  
 by the algorithm are  
 valid solutions  
 some are just  
 better than others

example suppose  $\bar{x}_{1_1} = 0.68$  and  $\bar{x}_{1_2} = 0.32$ ,  
 then  $\Pr [x_{1_1} = 1 \wedge x_{1_2} = 0] = 0.68$  and  $\Pr [x_{1_1} = 0 \wedge x_{1_2} = 1] = 0.32$

key idea: treat the solution to the  
 linear programming relaxation  
 as a probability distribution

Raghavan & Thompson (1987)  
 “Randomized rounding: a technique for provably  
 good algorithms and algorithmic proofs”



three simple ideas used in probabilistic proofs

### Linearity of Expectation

For any random variables  $X, Y$  and any  $a, b \in \mathbb{R}$ ,

$$\mathbb{E}[aX + bY] = a\mathbb{E}[X] + b\mathbb{E}[Y].$$

### Boole's inequality

Let  $\mathcal{E}$  be *any* set of events, then

$$\Pr[\bigvee_{E \in \mathcal{E}} E] \leq \sum_{E \in \mathcal{E}} \Pr[E],$$

regardless of any dependencies between the events.

### Chernoff's Bound

Let  $X = \sum_{j=1}^n X_j$ ,  $X_j \in \{0, 1\}$ , and  $\mu = \mathbb{E}[X]$ , then

$$\Pr[X > (1 + \epsilon)\mu] < \left( \frac{e^\epsilon}{(1 + \epsilon)^{(1+\epsilon)}} \right)^\mu < \exp(-\epsilon^2 \mu / 3)$$

What is the expected quality of the solution?

consider a vector  $s_i$  from  $\mathcal{F}$   
what is the expected distance between  $s_i$  and  $z$ ?

consider a row  $a_i$  from  $A$   
what is the expected value of  $a_i x$ ?

by our  
transformation from  
Minimum Covering Radius  
to the integer program, these  
questions are equivalent

we can answer these questions because we know the probabilities

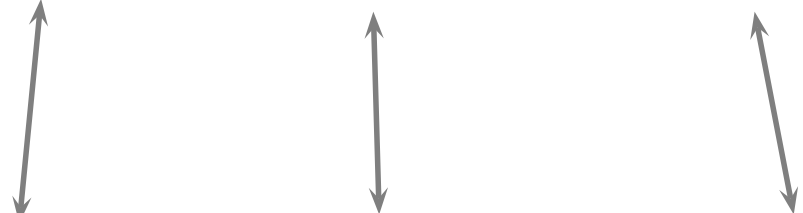
since  $\bar{x}$  was used as the probability when rounding

$$\begin{aligned}
 \mathbb{E}[d_H(s_i, z)] &= \mathbb{E}[a_i x] \\
 &= a_i \mathbb{E}[x] \\
 &= a_i \langle \mathbb{E}[x_{1_1}], \mathbb{E}[x_{1_2}], \dots, \mathbb{E}[x_{n_1}], \mathbb{E}[x_{n_2}] \rangle \\
 &= a_i \langle \bar{x}_{1_1}, \bar{x}_{1_2}, \dots, \bar{x}_{n_1}, \bar{x}_{n_2} \rangle \\
 &= a_i \bar{x}
 \end{aligned}$$

so we **expect** the randomly rounded solution vector to be of the same quality as the fractional solution, but **it could be better or worse**

What is the probability that the quality differs significantly from expectation?

the value of  $a_i x$  is a sum of 0 – 1 random variables **Bernoulli random variables**  
so we can apply Chernoff's Bound

$$\Pr [X \geq (1 + \epsilon)\mu] \leq \exp(-\epsilon^2 \mu / 3)$$

$$\Pr [a_i x \geq (1 + \epsilon)a_i \bar{x}] \leq \exp(-\epsilon^2 a_i \bar{x} / 3)$$

so now we have a bound on the probability  
that the product of  $x$  with **any individual constraint**  
exceeds  $1 + \epsilon$  times expectation

Hagerup & Rub (1989) "A guided tour of chernoff bounds"

How do we consider all rows simultaneously?

define the **bad** events

$$E_i \equiv "a_i x > (1 + \epsilon) a_i \bar{x}"$$

solution must avoid all  $E_i$

so we need to bound

$$\Pr(\bigvee_{i=1}^m E_i)$$

probability of violating at least one constraint

Boole's inequality says

$$\Pr(\bigvee_{i=1}^m E_i) \leq \sum_{i=1}^m \Pr(E_i)$$

combining with Chernoff's bound  
we require

$$\left( \frac{e^\epsilon}{(1 + \epsilon)^{(1+\epsilon)}} \right)^{a_i \bar{x}} < \frac{1}{2m}$$

for success with probability 1/2

$$\Pr(E_i) \leq \frac{1}{2m}$$

when we cannot bound the objective function from below

$$\epsilon \geq \frac{\ln(m)}{d}$$

$$\Pr(E_i) < \frac{1}{2m}$$

the probability of failure on  
a given attempt is  $< \frac{1}{2}$

conclusion      randomized rounding is a randomized  $O(\log m)$ -approximation algorithm  
for Minimum Covering Radius.

when we cannot bound the objective function from below

$$\epsilon \geq \frac{\ln(m)}{d}$$

$$\Pr(E_i) < \frac{1}{2m}$$

the probability of failure on  
a given attempt is  $< \frac{1}{2}$

conclusion randomized rounding is a randomized  $O(\log m)$ -approximation algorithm for Minimum Covering Radius.

with a large bound on the objective function

$$d \geq \frac{3 \ln(2m)}{\epsilon^2}$$

$$\Pr(E_i) < \frac{1}{2m}$$

the probability of failure on  
a given attempt is  $< \frac{1}{2}$

conclusion for any  $\epsilon > 0$  we can derive the number of repetitions required to guarantee a performance ratio of  $1 + \epsilon$ .

for **non-binary solution vectors**, we can often **use the fractional part** as the probability:

$$\text{Rounding up} \quad \Pr(x_i = \lceil \bar{x}_i \rceil) = \bar{x}_i - \lfloor \bar{x}_i \rfloor$$

$$\text{Rounding down} \quad \Pr(x_i = \lfloor \bar{x}_i \rfloor) = \lceil \bar{x}_i \rceil - \bar{x}_i$$

- ⌊ many NP-hard problems have fractional relaxations that are much easier than general linear programs

Fractional Knapsack reduces to sorting items in non-decreasing order of value/weight

- ⌊ the performance ratio is based on using the fractional relaxation as a lower bound  
exact solution is not required

a maximum matching algorithm can be used to obtain a fractional vertex cover  
that provides a lower bound on the optimal integral solution

The method can be (and has been) applied to a very wide range of problems  
and it has been shown to work well in practice.

→ Akamai uses a variant of randomized rounding for performance based load balancing

for Minimum Covering Radius any solution produced by randomized rounding was valid  
most integer programs are more difficult

a **packing problem** is the integer program  $\max\{cx \mid Ax \leq b\}$   
with the matrix and vectors  $A, b, c, x \geq 0$

**Note** unlike the minimum covering radius, packing problems  
have constraints that are not trivially satisfied  
because the right hand side  $b$  is fixed

## examples

Travelling Salesman	pack paths subject to degree 2 and cost constraints
Job Shop Scheduling	pack jobs subject to machine load constraints

## the point

it does not matter how good the solution is with respect to the objective function  
if a constraint  $a_i x \leq b_i$  is violated

randomly rounding can lead to violation of a constraint



the solution is to **scale the variables** used as probabilities, prior to rounding, so that the rounding process is likely to result in a feasible solution.

define the scaled vector  $\bar{x}' = \bar{x}/(1 + \epsilon)$  and use  $\bar{x}'$  instead of  $\bar{x}$  as the probabilities when rounding

analysis

$$\Pr(a_i x > (1 + \epsilon)a_i \bar{x}') = \Pr(a_i x > a_i \bar{x}) < \exp\left(\frac{\epsilon^2 a_i \bar{x}}{3(1 + \epsilon)}\right)$$

as before, the analysis proceeds using linearity of expectation, Chernoff's bound and Boole's inequality

to calculate the performance ratio we minimize  $\epsilon$  subject to

$$\frac{\epsilon^2}{(1 - \epsilon)} > \frac{2 \ln(m/4)}{a_i \bar{x}}$$

key idea: multiply the probabilities by some fraction less than 1

Raghavan & Thompson (1987)  
"Randomized rounding: a technique for provably good algorithms and algorithmic proofs"

## what is derandomization?

2 kinds of (algorithmic) derandomization

partial derandomization    reducing the number of coin flips (random bits) required  
often the goal is to produce better  
pseudorandom number generators

full derandomization    eliminate the use of random bits (no coin flips)  
sufficient to reduce random bits to  $O(\log n)$  for problem of size  $n$   
the number of sequences of  $O(\log n)$  random bits is bounded by  
a polynomial

### remark

complexity theoretic derandomization    proving that certain randomized complexity classes  
are equivalent to deterministic classes they are known  
to contain

similar to proving  $P=NP$ , but much more plausible, especially given  $PRIMES \in P$

if a particular coordinate is rounded to a specified value,  
is the probability of failure still  $< 1$ ?

with an oracle for the above question, we would not need random choices

define the event

$$\text{"fail"} \equiv \bigvee_{i=1}^m E_i$$

prior to the first random

$$\text{choice } \Pr(\text{fail}) < 1$$

the first decision  
requires selecting from  
2 possible choices

$$\Pr(\text{fail}) = \frac{\Pr(\text{fail} | x_{1_1} = 1) + \Pr(\text{fail} | x_{1_1} = 0)}{2}$$

$$\Rightarrow \min\{\Pr(\text{fail} | x_{1_1} = 1), \Pr(\text{fail} | x_{1_1} = 0)\} \leq \Pr(\text{fail})$$

if we knew the conditional probability  
we could make each choice  
so that  $\Pr(\text{fail})$  does not increase

$$\Pr(\text{fail} | \text{all choices made}) \in \{0, 1\}$$

since the probability of failure  
did not increase, all  $E_i$  have been avoided

key idea: use a probability function  
to guide a greedy algorithm

Erdos & Selfridge (1973) "On a combinatorial game"

the difficulty with conditional probabilities is computing them efficiently

the idea of a pessimistic estimator was introduced to characterize functions that can be used in the place of the true conditional probability

- 3 defining properties
- the estimator always bounds the probability of failure
  - the estimator initially indicates that probability of failure is not 1
  - the value of the estimator never increases as computation proceeds

known pessimistic estimators are often discovered by careful examination of the probabilistic proofs for the randomized algorithm  
pessimistic estimators hide in the proofs of bounds like Chernoff's

surprising observation

the **deterministic** algorithm has **performance** that is **better** than the expected performance of the **randomized** algorithm

Prabhakar Raghavan (1988)

"Probabilistic construction of deterministic algorithms: approximating packing integer programs"

very active area of research

two probabilistic ideas

- à correlated events (e.g. planarity and hamiltonicity of graphs)
- à events with limited dependence (e.g. coloring sparse graphs)

randomized algorithm      high probability  
 derandomized algorithm      positive probability

can we show positive probability  
 when high probability does not hold?

let  $N$  be a family of subsets of  $\{1, \dots, n\}$

$N$  is monotone decreasing if  $(X \in N) \wedge (X' \subseteq X) \Rightarrow (X' \in N)$

$N$  is monotone increasing if  $(X \in N) \wedge (X \subseteq X') \Rightarrow (X' \in N)$

Given a set  $N = \{1, \dots, n\}$  and some  $p \in [0, 1]^n$  suppose we select a random  $Y \subseteq N$  by placing each  $i \in N$  in  $Y$  independently with probability  $p_i$ . For any  $F \subseteq 2^N$ , let  $\Pr(F) \equiv \Pr(Y \in F)$ . Let  $F_1, F_2, \dots, F_s \subseteq 2^N$  be any sequence of monotone decreasing families.

$$\Pr \left( \bigwedge_{i=1}^s F_i \right) \geq \prod_{i=1}^s \Pr(F_i)$$

If we assume a symmetric probability space with  $\Pr = p$ , then  $\Pr \left( \bigwedge_{i=1}^s F_i \right) \geq p^s$ .

Fortuin, Ginibre & Kasteleyn (1971)  
 "Correlational inequalities for partially ordered sets"

- ▷ let  $N$  be the set of positions in the solution vector  $x$
- ▷ let the families  $F_i = \{S \subseteq N : a_i x \leq b\}$
- ▷ each family  $F_i$  corresponds to the set of solutions avoiding the bad event  $E_i$
- ▷ each  $F_i$  fits the definition of a monotone decreasing family

for **covering** and **packing** the events of constraint satisfaction are positively correlated  
we may treat them as though they are at least as probably as independent events

$$\begin{aligned}
 \Pr(\bigvee_{i=1}^m E_i) &\leq 1 - \Pr(\bigwedge_{i=1}^m \bar{E}_i) \\
 &\leq 1 - \Pr(\bigwedge_{i=1}^m F_i) \\
 &\leq 1 - p^m \\
 &< \textcolor{red}{1}
 \end{aligned}$$

$$\Pr(\bigvee_{i=1}^m X_i > (1 + \epsilon)\bar{X}) < \textcolor{red}{mp}$$

becomes

$$\Pr(\bigvee_{i=1}^m X_i > (1 + \epsilon)\bar{X}) < \textcolor{red}{p^m}$$

**positive probability** is too small for a randomized algorithm  
conditional probabilities can be used because a **pessimistic estimator** is known

Srinivasan (1995)

"Improved approximations of packing and covering problems"

two forms of the Lovasz Local Lemma: [general](#) and [symmetric](#)  
the general form is more powerful but difficult to use in algorithms

## Lovasz Local Lemma (Symmetric Form)

Let  $Y = \{y_1, \dots, y_m\}$  be events in an arbitrary probability space. Suppose each event  $y_i$  is mutually independent of all other events  $y_j$  except for at most  $d$ , and  $\Pr(y_i) \leq p$  for all  $1 \leq i \leq m$ . If  $ep(d+1) \leq 1$  then

$$\Pr\left(\bigwedge_{i=1}^m \bar{y}_i\right) > 0.$$

“ $e$ ” is the base of  
the natural logarithm

the symmetric form is easy to apply in existence proofs

[bonus](#) most objects shown to exist by the symmetric Lovasz Local Lemma  
can be found with a general algorithm known as

⌊ [Beck's hypergraph coloring algorithm](#)

J. Beck (1991) "An algorithmic approach to the lovasz local lemma I"

Erdos & Lovasz (1975)

"Problems and results on 3-chromatic hypergraphs and some related questions"



the Lovasz Local Lemma applied to integer programming

the events of interest  
are the  $E_i$  previously  
defined

$E_i$  and  $E_j$  are dependent  $\Leftrightarrow a_i$  and  $a_j$  share a variable

<u>example</u>	$a_1$	000011111100000000000000000000
$a_1$ and $a_3$	$a_2$	000000001111111000000000000000
are dependent	$a_3$	0000000000000000100111100000

Very high level overview of Randomized Rounding with Beck's Algorithm

- 1) do basic randomized rounding
- 2) remove all variables that do not appear in a violated constraint
- 3) remove all constraints for which all variables have been removed
- 4) form a graph on the set of remaining constraints with vertices corresponding to constraints and edges between constraints sharing a variable
- 5) **key idea** with high probability the connected components in the graph are small and can therefore be treated independently and efficiently

Chi-Jen Lu (1999)

"A deterministic approximation algorithm for a minimax integer programming problem"

---

Thank Prof. Horton and Prof. Evans for helping me with this course

Lots more, ask about applying these methods to your problems (especially routing problems)