Verjetnostne metode v računalništvu - zapiski s predavanj prof. Marca

Tomaž Poljanšek

Pregledal: Matija Kocbek

študijsko leto 2023/24

Kazalo

Poglavje 1

Introduction

1.1 Probability

(Ω, F, P_r):

- $\emptyset \in F$,
- $A \in F \implies A^c \in F$,
- $A_1, A_2 \dots \in F \implies \cup_{i=1}^{\infty} A_i \in F$.

$P_r(A) \geq 0$,
$P_r\left(\cup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P_r(A_i)$ if A_i disjoint,
$P_r\left(\cup_{i=1}^{\infty} A_i\right) \leq \sum_{i=1}^{\infty} P_r(A_i)$,
$\Omega = \{\omega_1, \omega_2 \dots\}$ - countable case.

$$\begin{pmatrix} \omega_1 & \omega_2 & \dots \\ p_1 & p_2 & \dots \end{pmatrix}$$

Example.

```
Alg():
  while True:
    B = sample as random from {0,1} # 1 with probability p
    if B = 1:
```

```
return
```

$\Omega = \{1, 01, 001, 0001 \dots\}$
$\begin{pmatrix} 1 & 01 & 001 & 0001 & \dots \\ p & (1-p)p & (1-p)^2p & (1-p)^3p & \dots \end{pmatrix}$.

1.2 Random variables

$X : \Omega \to \mathbb{Z}$.
$E[X] = \sum_{c\in\mathbb{Z}} c \cdot P_r(X = c)$: expected value of X.
Properties:

- $E[f(X)] = \sum_{c\in\mathbb{Z}} f(c) \cdot P_r(X = c)$,
- $E[aX + bY] = aE[X] + bE[Y]$,
- $E[X \cdot Y] = E[X] \cdot E[Y]$ if X, Y independent,
- $P_r(X \geq a) \leq \frac{E[X]}{a}$; $\forall a > 0\ \forall X \geq 0$ Markov inequality.

Example. (Continuing from before).
X = number of trials before return.
$X : \Omega \to \mathbb{Z}$.
$X : 1 \mapsto 1, 01 \mapsto 2, 003 \mapsto 3 \dots$
$\begin{pmatrix} 1 & 2 & 3 & 4 & \dots \\ p & (1-p)p & (1-p)^2p & (1-p)^3p & \dots \end{pmatrix}$ - geometric distribution.

Claim 1.2.1. $E[X] = \frac{1}{p}$.

Proof 1.2.2. $X = \sum_{i=1}^{\infty} X_i$.
$X_i = \begin{cases} 1: & \text{if trial } i \text{ is executed} \\ 0: & \text{else} \end{cases}$

$$\begin{aligned} E[X] = E\left[\sum_{i=1}^{\infty} X_i\right] = \sum_{i=1}^{\infty} E[X_i] = \\ = \sum_{i=1}^{\infty}(1-p)^{i-1} = \sum_{i=0}^{\infty}(1-p)^i = \frac{1}{1-(1-p)} = \frac{1}{p}. \end{aligned}$$

■

$E[X] = \frac{1}{p}$.

$P_r(X \geq 100 \cdot \frac{1}{p}) \overset{\text{Markov}}{\leq} \frac{E[X]}{\frac{1}{p}} = \frac{1}{100}$.

Definition 1.2.3. $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} = \sum_{i=1}^{\infty} \frac{1}{i}$.

Theorem 1.2.4. $H_n \leq 1 + \ln(n)$.

Proof 1.2.5.

$$H_n = 1 + \sum_{i=2}^{n} \frac{1}{i} \overset{\text{integral}}{\leq} 1 + \int_1^n \frac{dx}{x} = 1 + \ln(x)|_1^n = 1 + \ln(n).$$

■

Poglavje 2

Quicksort, min-cut

2.1 Quicksort

```
Input: set (no equal element) (unordered list) S ∈ ℝ
    (or whatever you can compare linearly)
Output: ordered list
Code:
  def Quicksort(S):
    if |S| = 0 or |S| = 1:
      return S
    else:
      a = uniformly at random from S
      S⁻ = {b ∈ S | b < a}
      S⁺ = {b ∈ S | a < b}
      return Quicksort(S⁻), a, Quicksort(S⁺)
```

$C(n)$ - random variable, the number of comparisons in evaluation of Quicksort with $|S| = n$.

Theorem 2.1.1. $E[C(n)] = O\left(N \log(n)\right)$.

Proof 2.1.2. $C(0) = C(1) = 0$.

$$
\begin{aligned}
E[C(n)] &= n-1+\sum_{i=1}^{n}\left(E[C(i-1)]+E[C(n-i)]\right)\cdot P_r(a \text{ is } i\text{-it element}) \leq \\
&\leq n+\frac{2}{n}\sum_{i=1}^{n-1}E[C(i)].
\end{aligned}
$$

Induction: (Inductive hypothesis is $C(n) \leq 5n\log n$)

$n = 1 : \checkmark$

$n-1 \to n$:

$$
\begin{aligned}
E[C(n)] &\leq n+\frac{2}{n}\sum_{i=1}^{n}E[C(i)] \leq \\
&\leq n+\frac{2}{n}\sum_{i=1}^{n}5i\log i \leq \\
&\leq n+\frac{2}{n}\sum_{i=1}^{\lfloor\frac{n}{2}\rfloor}5i\log i+\frac{2}{n}\sum_{i=1+\lfloor\frac{n}{2}\rfloor}^{n-1}5i\log i \leq \\
&\leq n+\frac{2}{n}\sum_{i=1}^{\lfloor\frac{n}{2}\rfloor}5i\log\frac{n}{2}+\frac{2}{n}\sum_{i=1+\lfloor\frac{n}{2}\rfloor}^{n-1}5i\log n \leq \\
&\overset{2.1}{\leq} n+\frac{2}{n}\left(\sum_{i=1}^{n}5i\log n-\sum_{i=1}^{\frac{n}{2}}5i\right) = \\
&= n+\frac{10}{n}\left(\frac{n(n-1)}{2}\log n-\frac{\frac{n}{2}(\frac{n}{2}.1)}{2}\right) \leq \\
&\leq n+5(n-1)\log n-n < \\
&< 5n\log n.
\end{aligned}
$$

$$
\log\frac{n}{2} = \log n - 1 \tag{2.1}
$$

■

This theorem gives us the following inequality: $P\left(C(n) \geq b\cdot 5n\log n\right) \overset{\text{Markov}}{\leq} \frac{1}{b}$. We give an alternative proof of the previous theorem:

Proof 2.1.3.

Let $S_1, S_2 \ldots S_n$ sorted elements of S.

Define random variable $X_{ij} = \begin{cases} 1: & \text{if } S_i \text{ and } S_j \text{ are compared} \\ 0: & \text{else} \end{cases}$

$C(n) = \sum_{1 \leq i < j \leq n} X_{ij}$.

$E[X_{ij}] = P(S_i \text{ and } S_j \text{ compared})$.

S_{ij} - the last set including S_i and S_j.

$E[X_{ij}] = \frac{2}{|S_{ij}|} \leq \frac{2}{j-i+1}$ because $|S_{ij}| \geq j - i + 1$ because S_{ij} contains all elements between S_i and S_j.

$$\begin{aligned} \implies E[C(n)] \leq \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1} = \\ \underset{\overline{\overline{}}}{k=j-i+1} \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{2}{k} \leq \\ \leq 2 \cdot n \cdot H_n \leq \\ \leq 2n(1 + \log n). \end{aligned}$$

■

2.2 Min-cut

G multigraph.

Cut: $U \subset V(G),\ U \neq \emptyset, V(g)$.

$(U, V(G) \setminus U) = \{uv \in E(G) \mid u \in U, v \in V(G) \setminus U\}$.

Problem min-cut:

Input: G.

Output: $\min |(U, V(G) \setminus U)|$ - cut size.

```
Algorithm 1:
   x ∈ V(G)
   Call maxFlow(G, x, y) ∀y ∈ V(G)
   Take min
```

maxFlow is Edmonds-Karp algorithm $O(|V||E|^2)$. This algorithm returns the minimum cut because maximum flow is equal to minimum cut.

A quicker deterministic algorithm is the Stoer-Wagner algorithm with time complexity $O\left(|E||V| + |V|^2 log|V|\right)$.

We give the probabilistic Karger algorithm:

```
Algorithm randMinCut:
  G_0 = G
  i = 0
  while |V(G_i)| > 2:
    e_i = uniformly at random from G_i (e_i is an edge)
    G_{i+1} = G_i/e_i (we contract e_i)
    i = i + 1
  u, v = V(G_{n-2}) # n = |V(G)|
  (u, v are vertices left after we finish the loop)
  U = {w ∈ V(G) | w is merged into u}
  return (U, V(G) \ U)
```

Theorem 2.2.1. Algorithm *randMinCut* gives you a minimal cut with probability greater or equal to $\frac{2}{n(n-1)}$.

Proof 2.2.2.
Fact 1: $minCut(G_i) \leq minCut(G_{i+1})$; Fact 2: $minCut(G) \leq \delta(G)$.
$k := minCut(G)$.
Let (A,B) be an optimal cut.
Let ε_i be the event when e_i not in (A,B).

$$
\begin{aligned}
&P_r(\text{Algorithm returning } (A,B)) \\
&= P_r(\varepsilon_0 \cap \dots \cap \varepsilon_{n-3}) \quad i = 0 \dots n-3 \\
&= P_r(\varepsilon_0 \cap \dots \cap \varepsilon_{n-4}) \cdot P_r(\varepsilon_{n-3} \mid \varepsilon_0 \cap \dots \cap \varepsilon_{n-4}) \\
&= P_r(\varepsilon_{n-3} \mid \cap_{i=0}^{n-4} \varepsilon_i) \cdot P_r(\varepsilon_{n-3} \mid \cap_{i=0}^{n-4} \varepsilon_i) \\
&\dots P_r(\varepsilon_1 \mid \varepsilon_0) \cdot P_r(\varepsilon_0) \\
&\overset{2.3}{\geq} \frac{n-2}{n} \cdot \frac{n-3}{n-1} \dots \frac{1}{3} = \frac{2}{n(n-1)}.
\end{aligned}
$$

$$P_r(\overline{\varepsilon_i} \mid \varepsilon_{i-1} \cap \cdots \cap \varepsilon_0) = \frac{k}{|E(G_i)|} \overset{2.2}{\leq} \frac{k}{\frac{(n-i)k}{2}} = \frac{2}{n-i}$$

$$|E(G_i)| \geq \frac{(n-i)\delta(G)}{2} \geq \frac{(n-i)k}{2}. \tag{2.2}$$

$$P_r(\varepsilon_i \mid \varepsilon_{i-1} \cap \cdots \cap \varepsilon_0) \geq 1 - \frac{2}{n-i} = \frac{n-2-i}{n-i}. \tag{2.3}$$

■

Theorem 2.2.3. Running *randMinCut* $n(n-1)$ times and taking best output gives correct solution with probability ≥ 0.86.

Proof 2.2.4. A_i - event that i-th run gives sub-optimal solution.

$$\begin{aligned} P_r(\text{solution not correct}) &= P_r(A_1 \cap \cdots \cap A_{n(n-1)}) \\ &= \prod_{i=1}^{n(n-1)} P_r(A_i) \leq \left(1 - \frac{2}{n(n-1)}\right)^{n(n-1)} \\ &\overset{2.4}{\leq} e^{-\frac{2}{n(n-1)} \cdot n(n-1)} = e^{-2} \leq 0.14. \end{aligned}$$

$$1 + x \leq e^x \; \forall x \in \mathbb{R}. \tag{2.4}$$

■

If we run *randMinCut* $n(n-1)\log(n)$ times, we get an incorrect answer with probability at most $\frac{1}{n^2}$ (repeat the previous proof). Because contracting an edge takes $m = |E|$ operations, we get the time complexity $O(mn^2 \log n)$. We could improve the algorithm to get time complexity $O\left(mn \log^3 n\right)$.

Poglavje 3

Complexity classes

Decision problem - yes/no question on a set of inputs = asking $w \in \Pi$.
Randomized algorithms:

- Las Vegas algorithms: always gives correct solution but isn't always quick, example: *Quicksort*.

- Monte Carlo algorithms: it can give wrong answers. Monte Carlo algorithms subtypes:

  – type(1): $\begin{cases} \omega \in \Pi \implies \text{alg. returns „}\omega \in \Pi\text{“ with probab. } \geq \frac{1}{2} \\ \omega \notin \Pi \implies \text{alg. returns „}\omega \in \Pi\text{“ with probab. } = 0 \end{cases}$

  – type(2): $\begin{cases} \omega \in \Pi \implies \text{alg. returns „}\omega \in \Pi\text{“ with probab. } = 1 \\ \omega \notin \Pi \implies \text{alg. returns „}\omega \in \Pi\text{“ with probab. } \leq \frac{1}{2} \end{cases}$

  – type(3): $\begin{cases} \omega \in \Pi \implies \text{alg. returns „}\omega \in \Pi\text{“ with probab. } \geq \frac{3}{4} \\ \omega \notin \Pi \implies \text{alg. returns „}\omega \in \Pi\text{“ with probab. } \leq \frac{1}{4} \end{cases}$

  type(1) and type(2): one-sided error, type(3): 2-sided error.
  $\frac{1}{2}$, $\frac{3}{4}$ and $\frac{1}{4}$ arbitrary numbers, can be something different (for type(3) better than coin flip).

Example. Decision problem: does a graph G have $minCut \leq k$?
Run $randMinCut(G)$ $n(n-1)$ times.

```
Algorithm randMinCut:
  if one of runs gives |(A,B)| ≤ k:
    return true
  else:
    return false
```

Complexity classes:

- RP (randomized polynomial time): decisional problems for which there exists Monte Carlo algorithm of type(1) with polynomial time complexity (worst case).
- co-RP: decisional problems for which there exists Monte Carlo algorithm of type(2) with polynomial time complexity (worst case).
- BRP (bounded-error probabilistic polynomial time): decisional problems for which there exists Monte Carlo algorithm of type(3) with polynomial time complexity (worst case).
- ZPP (zero-error probabilistic polynomial time): decisional problems for which there exists Las Vegas algorithm with expected polynomial time complexity (worst case).

ZPP = RP $\cap$ co-RP (proof: `https://cs.stackexchange.com/questions/54782/why-is-zpp-rp-%E2%88%A9-co-rp`)

Poglavje 4

Chernoff bounds

Theorem 4.0.1. Let $X_1, X_2 \dots X_n$ independent random variables with image $\{0, 1\}$. Let
$p_i = P_r(X_i = 1)$,
$X = \sum_{i=1}^{n} X_i$ and
$\mu = E(X) = p_1 + \dots + p_n$.
For every $\delta \in$ [0,1]:

$$P_r(X - \mu \geq \delta\mu) \leq e^{-\frac{\delta^2\mu}{3}}$$
$$P_r(\mu - X \leq \delta\mu) \leq e^{-\frac{\delta^2\mu}{2}}$$
$$\Longrightarrow P_r(|X - \mu| \geq \delta\mu) \leq 2e^{-\frac{\delta^2\mu}{3}}.$$

Probability falls extremely quickly after $E(X)$.

Proof 4.0.2.

$$
\begin{aligned}
P_r(X - \mu \geq \delta\mu) &= P_r(X \geq \mu(1+\delta)) \\
&\overset{t \geq 0}{=} P_r(tX \geq t\mu(1+\delta)) \\
&\overset{e^y \geq 0}{=} P_r(e^{tX} \geq e^{t\mu(1+\delta)}) \\
&\overset{\text{Markov}}{\leq} \frac{E\left(e^{tX}\right)}{e^{t\mu(1+\delta)}} \\
&\overset{4.1}{\leq} \frac{e^{(e^t-1)\mu}}{e^{t\mu(1+\delta)}} \\
&\overset{4.3}{\leq} e^{-\mu\frac{\delta^2}{3}}.
\end{aligned}
$$

$$
\begin{aligned}
E(e^{tX}) &= E(e^{tX_1+\cdots+tX_n}) \\
&= E(e^{tX_1}\dots e^{tX_n}) \\
&\overset{\text{independent}}{=\!=\!=} \prod_{i=1}^{n} E(e^{tX_i}) \\
&\overset{4.2}{\leq} \prod_{i=1}^{n} e^{p_i(e^t-1)} \\
&= e^{(e^t-1)\sum_{i=1}^{n} p_i} \\
&= e^{(e^t-1)\mu}.
\end{aligned}
\tag{4.1}
$$

$$
E(e^{tX_i}) = p_i \cdot e^t + (1-p_i)\cdot e^0 = 1 + p_i(e^t-1) \overset{2.4}{\leq} e^{p_i(e^t-1)}. \tag{4.2}
$$

We want:

$$
e^t - 1 - t(1+\delta) \leq -\frac{\delta^2}{3} \ \forall \delta \in (0,1) \tag{4.3}
$$

We select $t = \ln(1+\delta)$

$f(\delta) = 1 + \delta - 1 - (1+\delta)\ln(1+\delta) + \frac{\delta^2}{3} \overset{?}{\leq} 0$

$f(0) = 0$

$f'(\delta) = 1 - \ln(1+\delta) - 1 + \frac{2}{3}\delta = \frac{2}{3}\delta - \ln(1+\delta) \overset{?}{\leq} 0$

$\frac{2}{3}\delta \leq \ln(1+\delta)$(we prove this by showing that LHS - RHS has one stationary point on [0,1] and is negative in $\delta = 1$)

$\delta = 1:\ \frac{2}{3} \overset{?}{\leq} \ln(2) \approx 0.69\checkmark$

$$\begin{aligned} P_r(\mu - X \leq \delta\mu) &= P_r(X \geq \mu(1-\delta)) \\ &\overset{t\geq 0}{=} P_r(tX \geq t\mu(1-\delta)) \\ &\overset{e^y \geq 0}{=\!=} P_r(e^{tX} \geq e^{t\mu(1-\delta)}) \\ &\leq \cdots \leq \frac{e^{(e^t-1)\mu}}{e^{t\mu(1-\delta)}}. \end{aligned}$$

Want: $e^t - 1 - t(1-\delta) \leq -\frac{\delta^2}{2}\ \forall\delta \in (0,1)$:

We select $t = \ln(1-\delta)$

$f(\delta) = 1 - \delta - 1 - (1-\delta)\ln(1-\delta) + \frac{\delta^2}{2} \overset{?}{\leq} 0$

$f(0) = 0$

$f'(\delta) = -1 + 1 - \ln(1-\delta) + \delta \overset{?}{\leq} 0$

Similarly to before we prove that $\delta \leq \ln(1-\delta)$

■

Let now $X_i \sim \begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}$

$X = \sum_{i=1}^{n} X_i$

$\mu = \frac{n}{2}$

$$P_r\left(|X-\mu| \geq \sqrt{\frac{3}{2}n\ln(n)}\right) = P_r\left(|X-\mu| \geq \frac{n}{2}\sqrt{\frac{6}{n}\ln(n)}\right)$$
$$\overset{\text{Chernoff}}{\leq} 2e^{-\frac{\frac{n}{2}\frac{6}{n}\ln(n)}{3}} = \frac{2}{n};$$

For „big“ n is $\delta \in (0{,}1)$,
$\mu = \frac{n}{2}, \delta = \sqrt{\frac{6}{n}\ln(n)}$.
$d = \sqrt{\frac{3}{2}n\ln(n)}$

$$\implies P_r\left(X \in \left(\mu - \sqrt{\frac{3}{2}n\ln(n)}, \mu + \sqrt{\frac{3}{2}n\ln(n)}\right)\right) \geq 1 - \frac{2}{n}.$$

Claim 4.0.3.
Let $X_1, X_2 \dots$ independent random variables with image $\{0{,}1\}$.
$P_r(X_i = 1) = \frac{1}{2}\ \forall i$.
Let $X = \sum_{i=1}^{cm} X_i$ where $c \geq 4, m \in \mathbb{N}$.
Then $P_r(X \leq m) \leq e^{-\frac{cm}{16}}$.

Proof 4.0.4.

$$\begin{aligned} P_r(X \leq m) &= P_r\left(\frac{cm}{2} - X \geq \frac{cm}{2} - m\right) \\ &= P_r\left(\frac{cm}{2} - X \geq \frac{cm}{2}\left(1 - \frac{2}{c}\right)\right) \\ &\overset{\text{Chernoff}}{\leq} e^{-\frac{\frac{cm}{2}\left(1-\frac{2}{c}\right)^2}{2}} \\ &\overset{4.4}{\leq} e^{-\frac{\frac{cm}{2}\frac{1}{4}}{2}} = e^{-\frac{cm}{16}}. \end{aligned}$$

$$1 - \frac{2}{c} \geq \frac{1}{2} \text{ if } c \geq 4 \tag{4.4}$$

■

Back to Quicksort.

Theorem 4.0.5.
With probability $\geq 1 - \frac{1}{n}$ Quicksort uses at most $48n\ln(n)$ comparisons.

Proof 4.0.6.
Let $t_s - 1$ be the number of comparisons with s where s is not a pivot.. For $s \in S$ define $S_1, \dots, S_{t_s} \neq \emptyset$ sets that include s.
Define: iteration i is successful if $|S_{i+1}| \leq \frac{3}{4}|S_i|$ ($\frac{1}{2}$ is too strict).

$$X_i = \begin{cases} 1: & \text{if iteration } i \text{ is successful} \\ 0: & \text{else} \end{cases}$$

Notice: after at most $\log_{\frac{4}{3}}(n) = \frac{\ln(n)}{\ln(4)-\ln(3)}$ succesful iterations, the element s is sorted.
Probability that we haven't sorted s after $c\log_{\frac{4}{3}}(n)$ steps is at most:

$$\begin{aligned} P_r\left(\sum_{i=1}^{c\log_{\frac{4}{3}}(n)} X_i < \log_{\frac{4}{3}}(n)\right) &\leq P_r\left(\sum_{i=1}^{c\log_{\frac{4}{3}}(n)} Y_i < \log_{\frac{4}{3}}(n)\right) \\ &\overset{\text{Chernoff}}{<} e^{-\frac{c\log_{\frac{4}{3}}(n)}{24}} \\ &= e^{-\frac{c2\ln(n)\log_{\frac{4}{3}}(e)}{48}} \\ &= \frac{1}{n^2}\frac{c\log_{\frac{4}{3}}(e)}{48} \\ &\overset{4.6}{\leq} \left(\frac{1}{n}\right)^2. \end{aligned} \tag{4.5}$$

4.5 because X_i is not independent, we select such independent $Y_i \sim \begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}$, so that $X_i \geq Y_i$ (how do we construct such Y_i???)

$$\log_{\frac{4}{3}}(e) \approx 3.4, \text{ we select } c = 14. \tag{4.6}$$

Because of 4.6, we know that to sort s we will need at least $48\ln(n)$ iterations with probability less than $\left(\frac{1}{n}\right)^2$. Denote this event with A_s. The probability that we will need at least $48\ln(n)$ iterations to sort any $s \in S$ is $P(\bigcup_{s\in S} A_s) \leq \sum_{s\in S} P(A_s) \leq n\frac{1}{n^2} = \frac{1}{n}$. Thus we will need at $48n\ln(n)$ iterations in total with probability less than $\frac{1}{n}$ or in other words, total number of comparisons will be at most $48n\ln(n)$ with probability $1 - \frac{1}{n}$. ∎

Poglavje 5

Monte Carlo methods

5.1 Example 1

Area of circle $= \frac{\pi}{4}$.

$X_i = \begin{cases} 1: & \text{if you hit the area of circle} \\ 0: & \text{else} \end{cases}$

$P_r(X_i = 1) = \frac{\frac{\pi}{4}}{1} = \frac{\pi}{4}$.

$E(X_i) = \frac{\pi}{4}$.

$X = \frac{\sum_{i=1}^{n} X_i}{n}$.

$E(X) = \frac{n \cdot E(X_i)}{n} = E(X_i)$.

5.2 Example 2

$I = \int_\Omega f(x)dx$ - volume.

$X_i = \begin{cases} 1: & f(x_i,y_i) \le z_i \\ 0: & \text{otherwise} \end{cases}$

$v \cdot E\left(\frac{\sum_{i=1}^{n} X_i}{n}\right) = I$ where v is the volume of the subset from which we take random (x_i, y_i, z_i).

5.3 (ε,δ)-approximation

Definition 5.3.1 ((ε,δ)-approximation)**.** A random algorithm gives a (ε,δ)-approximation for value v if the output X satisfies:

$$P_r\left(|X - v| \leq \varepsilon v\right) \geq 1 - \delta.$$

Theorem 5.3.2. Let $X_1 \dots X_n$ be independent and identically distributed indicator variables. Let $\mu = E(X_i)$, $Y = \frac{\sum_{i=1}^{m} X_i}{m}$. If $m \geq \frac{3\ln\left(\frac{2}{\delta}\right)}{\varepsilon^2\mu}$, then $P_r\left(|Y - \mu| \geq \varepsilon\mu\right) \leq \delta \implies Y$ is (ε,δ)-approximation for μ.

Proof 5.3.3.
$X = \sum_{i=1}^{n} X_i$
$E(X) = mE(x_i) = m\mu$
$m \geq \frac{3\ln\left(\frac{2}{\delta}\right)}{\varepsilon^2\mu}$

$$\begin{aligned} P_r\left(|Y - \mu| \geq \varepsilon\mu\right) &= P_r\left(\left|\frac{X}{m} - \mu\right| \geq \varepsilon\mu\right) \\ &= P_r\left(\frac{1}{m}\left|X - E(X)\right| \geq \frac{1}{m}\varepsilon E(x)\right) \\ &\overset{\text{Chernoff}}{\leq} 2e^{-\frac{\varepsilon^2 E(x)}{3}} \\ &= 2e^{-\frac{\varepsilon^2\mu m}{3}} \\ &\leq 2e^{-\frac{\varepsilon^2\mu}{3}\cdot\frac{3\ln\left(\frac{2}{\delta}\right)}{\varepsilon^2\mu}} = \delta. \end{aligned}$$

■

Back to example 1:
$E(Y) = \frac{\pi}{4}, \delta = \frac{1}{1000}$ (99.9% sure), $\varepsilon = \frac{1}{10000}$
$\implies M = \frac{3\ln\left(\frac{2}{\frac{1}{1000}}\right)4}{\pi\left(\frac{1}{10000}\right)^2} \approx 29106.$
Problems for MC (Monte-Carlo):

- rare events, e.g. $X \sim \begin{pmatrix} 0 & 10^{100} \\ 1 - 10^{-20} & 10^{-20} \end{pmatrix}$, $E(X) = 10^{80}$

5.4 DNF counting

CNF: $(X_{i_1} \vee \overline{X_{i_2}} \vee X_{i_4}) \wedge (X_{i_1} \vee \overline{X_{i_3}}) \wedge \dots$
DNF: $(\overline{X_{i_1}} \wedge X_{i_2} \wedge \overline{X_{i_4}}) \vee \dots$ - easy to determine if solution exists - this is the negation of the above CNF.
Question: number of solutions to a given DNF?
Observation: CNF F has a solution $\iff$ DNF $\neg F$ has less than 2^n solutions, n is number of samples.

```
ALG_1(F):
  x = 0
  for i in range(1, m+1):
    x_1 ... x_n uniformly random from {0,1}^n
    if F(x_1 ... x_n) = 1:
      x += 1
  return x/m · 2^n
```

$$X_i = \begin{cases} 1: & F(x_{i1}, \dots, x_{in}) = 1 \\ 0: & \text{otherwise} \end{cases}$$
$Y = \frac{\sum_{i=1}^m X_i}{m}$
We attempt to achieve an (ε,δ)-approximation for the number of solutions of the DNF using Y.
$E(X_i) = P_r(X_i = 1) = \frac{\text{number of solutions of } F}{2^n}$
$E(Y) = \frac{\text{number of solutions of } F}{2^n} = \frac{c(F)}{2^n}$
For Y to be an (ε,δ)-approximation we need $m \geq \frac{3\ln\left(\frac{2}{\delta}\right)}{\varepsilon^2 E(X)} = \frac{3\ln\left(\frac{2}{\delta}\right)}{\varepsilon^2} \cdot \frac{2^n}{c(F)}$ according to the previous theorem.
$c(F)$ very small $\to$ m exponentially big $\to$ not good (we need a lot of samples).

Definition 5.4.1.
$SC_i = \{(a_1 \dots a_n) \in \{0,1\}^n \text{ such that } F = F_1 \vee \dots \vee F_t,\ F_i(a_1 \dots a_n) = 1\}$.

Suppose F_i is a conjuction of l_i different variables $a_{i_1}, \dots, a_{i_{l_i}}$ (or their ne-

gations). If $F_i(a_1, \ldots, a_n) = 1$, this uniquely determines $a_{i_1}, \ldots, a_{i_{l_i}}$ and so $|SC_i| = 2^{n-l_i}$
$U = \{(i,a) \mid i \in \{1,2\ldots t\}, a \in SC_i\}$
$|U| = \sum_{i=1}^{t} |SC_i|$ (space smaller than $\{0,1\}^n$)
$S = \{(i,a) \in U \mid a \in SC_i,\ a \notin SC_j\ 1 \leq j < i\}$
$|S| = c(F)$ (each solution appears exactly once in S).

```
ALG_2(F):
  x = 0
  for i in range(1,m+1):
    (i,a) uniformly random from U
    if (i,a) ∈ S:
      x += 1
  return x/m · |U|
```

In order to check whether $a \in S$, we need to check whether there is such i so that $a \in SC_i$ and $a \notin SC_j\ j = 1 \ldots i-1$. To check whether or not $a \in SC_i$ takes $O(n)$ time, so the it takes $O(tn)$ time to check whether $a \in S$.

We can implement uniform random selection from U with $O(1)$ time complexity so the total time complexity is $O(t \cdot n \cdot m)$.

Theorem 5.4.2. For $m = \left\lceil \frac{3t\ln\left(\left(\frac{2}{\delta}\right)\right)}{\varepsilon^2} \right\rceil$ algorithm returns (ε,δ)-approximation in $O\left(\frac{t^2 n \ln\left(\frac{2}{\delta}\right)}{\varepsilon^2}\right)$ time.

The time complexity follows immediately from the previous analysis. Define

$$X_i = \begin{cases} 1: & i\text{-th selection is in S} \\ 0: & \text{otherwise} \end{cases}$$

$$Y = \frac{\sum_{i=1}^{m} X_i}{m}$$

We have $c(F) = |S|, E(Y) = \frac{|S|}{|U|}$ and so

$$P_r(Y|U| - c(F) > \varepsilon c(F)) = P_r(|U|(Y - E(Y)) > \varepsilon|U|E(Y)) \leq \delta$$

if

$$m \geq \frac{3\ln\left(\frac{2}{\delta}\right)}{\varepsilon^2 E(Y)}.$$

We have

$$E(Y) = \frac{|S|}{|U|} \geq \frac{1}{t}$$

This is because SC_i contains at most $c(F)$ elements for every i and $|S| = c(F)$.

So it suffices to take

$$m \geq \frac{3\ln\left(\frac{2}{\delta}\right)t}{\varepsilon^2} \geq \frac{3\ln\left(\frac{2}{\delta}\right)}{\varepsilon^2 E(Y)}.$$

In new space $E(Y)$ much larger $\implies$ m smaller.

Poglavje 6

Polynomials

Let $\mathbb{F}$ be a field.
$\mathbb{F}$ can be $\mathbb{R}, \mathbb{C}, \mathbb{Z}_p, \mathbb{F}_{p^n}$.
$\mathbb{F}[x_1 \dots x_n]$ algebra of polynomials with values $x_1 \dots x_n$.
$f \in \mathbb{F}[x_1 \dots x_n]$
$deg(f[x_1 \dots x_n]) := deg(f[x \dots x])$.

Theorem 6.0.1. Let $p(x_1 \dots x_n) \in \mathbb{F}[x_1 \dots x_n]$ have the degree $d \geq 0$ and $p \neq 0$. Let $S \subset \mathbb{F}$ be finite. If $(r_1 \dots r_n)$ is uniformly at random element from S^n. Then $P_r(p(r_1 \dots r_n) = 0) \leq \frac{d}{|S|}$.

Proof 6.0.2. Induction on n.
$n = 1$:

$$\begin{aligned}
&p(x) = (x - z_1)(x - z_2) \dots (x - z_j) q(z) \\
&\text{Fact: number of zeros } \leq \text{ degree} \\
&P_r(p(r_1) = 0) = \frac{\text{number of zeros in } S}{|S|} \leq \frac{d}{|S|}.
\end{aligned}$$

$n-1 \to n$:

$$\begin{aligned}
&\text{rewrite } p: \\
&p(x_1 \dots x_n) = \sum_{i=0}^{j} x^i p_i(x_2 \dots x_n) \\
&\text{for some } j \leq d
\end{aligned}$$

$$\begin{aligned}
P_r(p(r_1 \dots r_n) = 0) &= P_r(p(r_1 \dots r_n) = 0 \mid p_j(r_2 \dots r_n) = 0) \cdot P_r(p_j(r_2 \dots r_n) = 0) \\
&+ P_r(p(r_1 \dots r_n) = 0 \mid p_j(r_2 \dots r_n) \neq 0) \cdot P_r(p_j(r_2 \dots r_n) \neq 0) \\
&\leq 1 \cdot \frac{d-j}{|S|} + \frac{j}{|S|} \cdot 1 \\
&= \frac{d}{|S|},
\end{aligned}$$

because

$$P_r(p_j(r_2 \dots r_n) = 0) \leq \frac{d-j}{|S|} \; (\deg(p_j) \leq d-j \text{ because otherwise } \deg(p) > d)$$

$$P_r(p(r_1 \dots r_n = 0) \mid p_j(r_2 \dots r_n) \neq 0) \leq \frac{j}{|S|} \; \text{(WHY???)}.$$

■

Problem:
Let A,B,$C \in \mathbb{F}^{n \times n}$, is $A \cdot B = C$?
Computing $A \cdot B$:

- school-book algorithm: $O(n^3)$,
- Strassen algorithm: $O(n^{2{,}807\dots})$,
- galactic algorithm: $O(n^{2.372\dots})$ - has enormous constants.

```
RAND_ABC(A,B,C):
  for i in range(1,k+1):
    x uniformly at random from {0,1}^n
    if A·(B·x) ≠ C·x:
      return false
```

```
    return true
```

The time complexity of the above algorithm is $O(kn^2)$.

If $A \cdot B = C$, algorithm returns true.
If $A \cdot B \neq C$:

$$\begin{aligned} P_r(ABx = Cx) &= P_r((AB - C)x = 0) \\ &= P_r\left(||(AB - C)x||^2 = 0\right) \overset{\text{Poly}}{\leq} \frac{2}{3}. \end{aligned}$$

$||(AB-C)x||^2$ - polynomial in $x_1 \ldots x_n$ of degree 2.
If $A \cdot B \neq C$, then algorithm return false with probability at least $1 - \left(\frac{2}{3}\right)^k$.
Problem:
1-factor in bipartite graphs.
$|V(g)| = 2n$.
Represent G with $n \times n$ matrix $Z = (Z_{ij})_{i,j=1}^n$

$$Z_{ij} = \begin{cases} X_{ij}: & \text{if } a_i b_j \in E(x) \qquad \text{(X: variable)} \\ 0: & \text{else} \end{cases}$$

$$\begin{aligned} detZ(x_{11} \ldots x_{nn}) &= \sum_{\pi \in S_n} sign(\pi) z_{1,\pi(1)} \ldots z_{n,\pi(n)} \\ &= \sum_{\pi \in S_n, \pi \text{ defines 1-factor}} sign(\pi) x_{1,\pi(1)} \ldots x_{n,\pi(n)}. \end{aligned}$$

$detZ \neq 0 \iff G$ has 1-factor.

```
  Rand_1factor(G):
    construct Z with variables x_11 ... x_nn
    for i in range(1,k+1):
      u <- uniformly at random from {1,2..2n-1}^(n^2) (r_{11} ... r_{nn
      compute d = detZ(r_11...r_nn)
      if d != 0:
        return true
    return false
```

Complexity: $k\cdot$ computing determinant: $O\left(n^3\right)$ (Gaussian elimination).
or apply approximation algorithm:

- if G has no 1-factor it always returns false,
- if G has 1-factor, it returns true with probability at least $1-\left(\frac{n}{2n}\right)^k = 1-\left(\frac{1}{2}\right)^k$ (k konstant, larger set $\implies$ smaller k needed).

Poglavje 7

Random graphs

7.1 G(n,p) model

G is a random Erdös-Rény graph if it has n vertices and each pair of vertices is connected with probability p.

Example. $G\left(5, \frac{1}{2}\right)$.

$E(\text{edges in } G \text{ from } G(n,p)) = \sum_{1 \le i < j \le n} E(X_{ij}) = \binom{n}{2} p$.

$X_{ij} = \begin{cases} 1: & \text{if } i \text{ and } j \text{ have edge} \\ 0: & \text{otherwise} \end{cases}$

p can be function of n.

Y_v : degree of v.

$E(Y_v) = (n-1)p$.

Definition 7.1.1.

We say that a random graph has some property almost surely (A.S.) if
$P_r(G \in G(n,p) \text{ has property}) \xrightarrow{n \to \infty} 1$.

Claim 7.1.2.

Let p be constant. Then $G \in G(n,p)$ has diameter 2 A.S.

Proof 7.1.3.

Let $u,v \in V(G)$

$X_w = \begin{cases} 1: & \text{if } uw \in E(G) \text{ in } vw \in E(G) \\ 0: & \text{else} \end{cases}$

$P_r(X_w = 1) = p^2$

$P_r(X_w = 0 \text{ for all } w \neq u,v) = (1-p^2)^{n-2}$.

$$\begin{aligned} P_r(G \text{ has diameter} > 2) &= P_r(X_w = 0 \text{ for all } w \notin u,v \text{ for some } u,v) \\ &\leq \binom{n}{2}(1-p^2)^{n-2} \overset{n\to\infty}{\longrightarrow} 0 \end{aligned}$$

$\binom{n}{2}$ - polynomial, $e^{\cdots}$ - exponent. ■

$p = f(n)$

$\frac{1}{n}, \frac{1}{n^3}, \frac{\log n}{n}$

Theorem 7.1.4. (without proof)

Let p be a function of n, let $G \in G(n,p)$:

- $np < 1 \implies G$ A.S. disconnected with connected components of size $O(\log n)$,
- $np = 1 \implies G$ A.S. has 1 large component of size $O\left(n^{\frac{2}{3}}\right)$,
- $np = c > 1 \implies G$ A.S. has giant component of size dn, $d \in (0,1)$,
- $np \leq (1-\varepsilon)\ln n \implies G$ A.S. disconnected with isolated vertices,
- $np > (1-\varepsilon)\ln n \implies G$ A.S. connected.

Theorem 7.1.5.

Let $np = \omega(n)\ln(n)$ for $\omega(n) \to \infty$ „very slowly“ think of $\omega(n) = \log(\log n)$, then $diam(G)$ in $\Theta\left(\frac{\ln n}{\ln(np)}\right)$ for G in $G(n,p)$.

Lemma 7.1.6.

Let $S \subset V(G), |S| = cn$ for $c \in (0,1]$ and $v \notin S$.

then $cnp(1-\omega^{-\frac{1}{3}}) \leq N_S(v) \leq cnp(1+\omega^{-\frac{1}{3}})$ A.S. ($\omega^{-\frac{1}{3}} \to 0$ very slowly).

Proof 7.1.7. (Lemma):
$E(N_s(v)) = c \cdot n \cdot p, \delta = \omega^{-\frac{1}{3}}$

$$
\begin{aligned}
P_r(|N_s(v) - cnp| \geq \delta cnp) &\overset{\text{Chernoff}}{\leq} 2e^{-\frac{\omega^{-\frac{2}{3}} cnp}{3}} \\
&= 2e^{-\frac{cnp}{3\omega(n)^{\frac{2}{3}}}} \xrightarrow{n \to \infty} 0.
\end{aligned}
$$

For all v: $n \cdot 2e^{-\frac{cnp}{3\omega(n)^{\frac{2}{3}}}} \xrightarrow{n \to \infty} 0.$ ∎

Proof 7.1.8. (Theorem):
k be such that $\sum_{i=0}^{k-1} |N_i| \leq \frac{n}{2}, \sum_{i=0}^{k} |N_i| > \frac{n}{2}$.
$|N_0| = 1$
$|N_i| \leq |N_{i-1}| \cdot n \cdot p \cdot (1 + \omega^{-\frac{1}{3}})$:
$|S| \leq n,\ np(1 + \omega^{-\frac{1}{3}})$-each element.
$k \overset{?}{=} \frac{\log\left(\frac{n}{3}\right)}{\log\left(n \cdot p \cdot \left(1 + \omega^{-\frac{1}{3}}\right)\right)} = \log_{np(1+\omega^{-\frac{1}{3}})} \frac{n}{3} = \Theta\left(\frac{\ln(n)}{\ln(np)}\right)$.
$|N_{\leq k}| = |N_1 \cup \cdots \cup N_k|$.

$$
\begin{aligned}
|N_{\leq k}| &\leq \sum_{i=0}^{k} (np(1 + \omega^{-\frac{1}{3}}))^i \\
&= \frac{(np(1 + \omega^{-\frac{1}{3}}))^{k+1} - 1}{np(1 + \omega^{-\frac{1}{3}}) - 1} \\
&< \frac{np(1 + \omega^{-\frac{1}{3}})^{k+1}}{\frac{1}{2} np(1 + \omega^{-\frac{1}{3}})} \\
&= 2np(1 + \omega^{-\frac{1}{3}})^k \\
&\overset{k}{=} 2 \cdot \frac{n}{3} \text{ - haven't covered all} \\
&\implies diam(G) > k \text{ bound from below.}
\end{aligned}
$$

$N_i \subseteq S$
$\frac{1}{2} np \left(1 - \omega^{-\frac{1}{3}}\right) \cdot |N_{i-1}| \leq |N_i|$

$$
\begin{aligned}
n &\geq \sum_{i=0}^{k} |N_i| \\
&\geq \sum_{i=0}^{k} \left(\frac{1}{2}np\left(1-\omega^{-\frac{1}{3}}\right)\right)^i \\
&= \frac{\left(\frac{1}{2}np\left(1-\omega^{-\frac{1}{3}}\right)\right)^{k+1}-1}{\frac{1}{2}np\left(1-\omega^{-\frac{1}{3}}\right)-1} \\
&\geq \left(\frac{1}{2}np\left(1-\omega^{-\frac{1}{3}}\right)\right)^k \qquad / \ln
\end{aligned}
$$

$\frac{\ln n}{\ln(np)} \approx \frac{\ln n}{\ln\left(\frac{1}{2}np\left(1-\omega^{-\frac{1}{3}}\right)\right)} \geq k$.
$\implies w \in S'$.
Number of neighbors in N_k A.S. ≥ 1,
$|N_k| \geq \left(\frac{1}{2}np\left(1-\omega^{-\frac{1}{3}}\right)\right)^k \approx c \cdot n$
$\implies diam(G) = k+1$ A.S. ■

7.1.1 Scale free property

$G \in G(n{,}p)$.

In real world: $p(k)$ = proportion of degree k vertices.

$\log(p(k)) = -\gamma \cdot \log k$

$p(k) = k^{-\gamma}$.

Internet: $\gamma \approx 3.42$,

protein reactions: $\gamma \approx 2.89$.

7.2 Barbási-Albert Model

B.A. model.

Start with m modes.

Grow:

- add node v,

- add m edges from v (to u),
- for each new edge: $P(v \sim u) = \frac{degu}{\sum_x degx}$.

Theorem 7.2.1.

B.A. model has scale free property, in particular
$p_k = \frac{2m(m+1)}{k(k+1)(k+2)}$.

Definition 7.2.2.

$p_n(k)$: expected proportion of degree k vertices in graph with k vertices,
$p_k := \lim_{n\to\infty} p_n(k)$.

Proof 7.2.3.

$p_n(k) \cdot n$: expected number of degree k vertices,
$p_n(k)n \cdot \frac{k}{\sum_u degu} m = p_n(k) \cdot \frac{k}{2}$: expected number of degree k vertices changing into degree $k+1$ vertices.
$\sum_u degu = 2|E|$
$p_{n+1}(k) \cdot (n+1) = p_n(k) \cdot n - p_n(k) \cdot \frac{k}{2} + p_n(k-1) \cdot \frac{k-1}{2}$, where
$p_n(k) \cdot n$: degree $k \to k$,
$p_n(k) \cdot \frac{k}{2} : k \to k+1$,
$p_n(k-1) \cdot \frac{k-1}{2} : k-1 \to k$.
For n very big (very close to limit):
$p_k \cdot (n+1) = p_k \cdot n - p_{k-1} \cdot \frac{k}{2} + p_{k-1} \cdot \frac{k-1}{2}$
$\implies p_k = \frac{k-1}{k+2} p_{k-1}$.
For degree m:
$(n+1) \cdot p_{n+1}(m) = p_n(m) \cdot n - p_n(m) \cdot \frac{m}{2} + 1$
$(n+1) \cdot p_m = n \cdot p_m - \frac{m}{2} \cdot p_m + 1$

$$\begin{aligned} p_m &= \frac{2}{m+2} \\ \implies p_{m+1} &= \frac{2}{m+2} \cdot \frac{m}{m+3} \\ \implies p_{m+2} &= \frac{2m(m+1)}{(m+2)(m+3)} \\ \implies p_k &= \frac{2m(m+1)}{k(k+1)(k+2)}. \end{aligned}$$

Poglavje 8

Markov chains

Ω: finite set (of states).

Definition 8.0.1 (Markov chain)**.**
(Discrete time) Markov chain is a sequence of random variables $X = X_0, X_1, X_2 \ldots$ with image Ω and properties:

- $P(X_{i+1} = x \mid X_i = x_i, X_{i-1} = x_{i-1} \ldots X_0 = x_0) =$
  $P(X_{i+1} = x \mid X_i = x_i)$ - Markov property,

- $P(X_{i+1} = x \mid X_i = y) = P(X_1 = x \mid X_0 = y)$ - time is homogenous.

Example.
$\Omega = \mathbb{Z}_5$
$P(X_{i+1} = x + 1 \mid X_i = x) = \frac{1}{2}$
$P(X_{i+1} = x - 1 \mid X_i = x) = \frac{1}{2}$.

Definition 8.0.2 (Transition matrix)**.**
$\Omega = \{x_1 \ldots x_n\}$
$p_{ij} = P(X_{t+1} = j \mid X_t = i)$

$$\begin{bmatrix} p_{11} & \dots & \\ p_{1n} & & \\ \vdots & & \vdots \\ p_{n1} & \dots & p_{nn} \end{bmatrix}.$$

Definition 8.0.3 (Transition graph)**.**
Edge between states i and j exists if $p_{ij} > 0$.

P is stochastic matrix:
$p_{ij} \in [0,1]$
$\sum_j p_{ij} = 1$ (row sum).
We choose beginning state randomly.
$q(0) = (q_1(0) \dots q_n(0))$
$P(X_0 = i) = q_i(0)$.
Let $q(t) = (q_1(t) \dots q_n(t))$
$P(X_t = i) = q_i(t)$.
It holds: $q(t) = q(t-1) \cdot P = q(0) \cdot P^t$.

$$\begin{bmatrix} 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$q(0) = (1, 0, 0, 0, 0)$
$q(1) = (1, \frac{1}{2}, 0, 0, \frac{1}{2})$
$q(2) = (\frac{1}{2}, 0, \frac{1}{4}, \frac{1}{4}, 0)$
$\vdots$

Definition 8.0.4.

- Distribution π is stationary if $\pi = \pi \cdot P$,
- f_{ij}: probability that $X_t = x_j$ for some t assuming $X_0 = x_i$,

- h_{ij}: expected number of steps needed to get to state X_j strting in X_i (hitting time),

- $N(i,t,q(0))$: expected number of times we visit x_i after t steps starting with distribution $q(0)$,

- $\forall f_{ij} > 0 \iff$ transition graph is strongly connected $\iff$ we say the chain is irreducible,

- M.C. is aperiodic if there is no $c \in \{2,3,4\dots\}$ such that all lengths of cycles are divisible by c.

Theorem 8.0.5.
Let X be finite irreducible M.C. Then:

a) there exists unique stationary distribution $\pi = (\pi_1 \dots \pi_n)$,

b) $f_{ii} = 1, h_{ii} = \frac{1}{\pi_i}$,

c) $\lim_{t\to\infty} \frac{N(i,t,q(0))}{t} = \pi_i$ - approaches π regardless of $q(0)$,

d) if X is aperiodic: $\lim_{t\to\infty} q(0) \cdot P^t = \pi$.

Example.

$$P = \begin{bmatrix} 0 & \frac{1}{2} & \dots & \frac{1}{2} \\ \frac{1}{2} & 0 & \dots & 0 \\ \vdots & & & \vdots \\ \dots & & \frac{1}{2} & 0 \end{bmatrix}$$

$\pi = (\frac{1}{n} \dots \frac{1}{n})$

$h_{i,i} = n$

$n = h_{i,i} = 1 + \frac{1}{2}h_{i-1,i} + \frac{1}{2}h_{i+1,i}, \quad h_{i-1,i} = h_{i+1,i}$

$n - 1 = h_{i-1,i}$

$E(\text{steps around}) \leq h_{0,1} + h_{1,2} + \dots + h_{n-1,n} \leq n(n-1).$

8.1 2-SAT

Recall: k-SAT:

$$F = C_1 \wedge \cdots \wedge C_m$$
$$C_i = X_{i1} \vee \cdots \vee X_{ik}.$$

3-SAT: NP complete.

```
Algorithm:
  def rand2SAT(F):
    b^0 = (b_0^0 ... b_n^0)
    for i in range(t):
      if F(b^i) = 1:
        return True
      C_l <- clause that is False
      x_j <- uniformly at random from x_l1 and x_l2
      b^(i+1) = (b_0^i ... overline(b_j^i) ... b_n^i)
    if F(b^t) = 1:
      return True
    return False
```

Theorem 8.1.1.
If $k = 8n^2$, then $P(\text{rand2SAT} = \text{True} \mid \text{correct answer is True}) \geq \frac{3}{4}$.

Proof 8.1.2.
Let $a = (a_1 \dots a_n)$ be a correct solution.
Let X_i = Hamming distance from b^i to a.
Goal: bound $h_{n,0}$.
$P(\text{distance of } b^{i+1} \text{ to } a \text{ is } j-1 \mid \text{distance of } b^i \text{ to } a \text{ is } j) \geq \frac{1}{2}$.

$$P = \begin{bmatrix} 0 & 1 & \dots & 0 \\ \frac{1}{2} & 0 & \dots & 0 \\ \vdots & & & \vdots \\ \dots & & 1 & 0 \end{bmatrix}$$

$\pi \stackrel{?}{=} \pi P$
$\pi = (\frac{1}{2n}, \frac{1}{n} \dots \frac{1}{n}, \frac{1}{2n})$
By theorem
$h_{i,i} = \frac{1}{\pi_i} = n$ for $i = 1, 2 \dots n-1$
$h_{0,0} = h_{n,n} = 2n$
$n = h_{i,i} = 1 + \frac{1}{2}h_{i+1,i} + \frac{1}{2}h_{i-1,i}$
$h_{i+1,i} \leq 2n$
$i = 0 :\ 2n = h_{0,0} = 1 + h_{1,0} \implies h_{1,0} < 2n$
$h_{n,0} \leq h_{n,n-1} + \dots + h_{1,0} \leq 2n^2$
$E(\text{steps in algorithm to reach correct solution}) = E(Z) \leq 2n^2$
$P(\text{algorithm hasn't reached correct solution after } 8n^2 \text{ steps})$
$= P(Z > 8n^2) \stackrel{\text{Markov}}{\leq} \frac{E(Z)}{8n^2} \leq \frac{1}{4}.$ ■

8.2 Generating a uniformly random element of a set

Ω: set.
Let G be a symmetric graph on Ω.
We form M.C:

$$P_{x,y} = \begin{cases} \frac{1}{M} : & \text{if } x \neq y \wedge x \sim y \\ 0 : & \text{if } x \neq y \wedge x \nsim y \\ 1 - \frac{|N(x)|}{M} : & \text{if } x = y \end{cases}$$

$M \geq \max_{v \in \Omega} |N(v)|$.
If G is connected $\implies$ M.C. is irrecudible.
$\pi = (\frac{1}{|\Omega|} \dots \frac{1}{|\Omega|})$
$\pi \stackrel{?}{=} \pi P$

$$
\begin{aligned}
(\pi P)_x &= \sum_y \pi_y P_{y,x} \\
&= \sum_{y \in N(x)} \frac{1}{M} \cdot \frac{1}{|\Omega|} + \frac{1}{|\Omega|}\left(1 - \frac{|N(x)|}{M}\right) = \frac{1}{|\Omega|} = \pi_x.
\end{aligned}
$$

$\implies$ if we walk on the Markov chain long enough, we end up in state x with probability $\pi_x = \frac{1}{|\Omega|}$
$\implies$ we can sample uniformly.

Example.
G graph, finding largest independent set ($\forall u,v : u \nsim v$) is NP-complete.
Lets try sampling a uniformly random independent set
$\Omega = \{\text{independent sets}\}$
$u \sim v$ if $|u \triangle v| = 1$ $((u \cup \{el\}) = v)$
M.C.: X_0 = arbitrary independent set
X_{i+1}:

- pick uniformly at random $v \in V(G)$,
- if $v \in U$ then $X_{i+1} = U \backslash \{v\}$,
- if $U \cup \{v\}$ is independent then $X_{i+1} = U \cup \{v\}$,
- else $X_{i+1} = U$.

M is number of vertices
$\implies$ $\forall u \in \Omega : \lim_{t\to\infty} P(X_t = u) = \frac{1}{|\Omega|}$.
Note: irredudicle; $U \to \emptyset \to V$, aperiodic.

8.3 Metropolis algorithm

Ω: set,
π: chosen distribution on Ω.
Make G graph on Ω

$$P_{x,y} = \begin{cases} \frac{1}{M} \cdot \min\left(1, \frac{\pi_y}{\pi_x}\right) : & \text{if } x \neq y \wedge x \sim y \\ 0 : & \text{if } x \neq y \wedge x \nsim y \\ 1 - \sum_{y \in N(x)} : & \text{if } x = y \end{cases}$$

$M \geq \max_{v \in \Omega} |N(v)|$

$\pi \stackrel{?}{=} \pi P$

$$\begin{aligned}(\pi P)_x &= \sum_y \pi_y P_{y,x} = \sum_{y \in N(x)} \pi_y \frac{1}{M} \min\left(\left(1, \frac{\pi_y}{\pi_x}\right)\right) + \pi_x \left(1 - \sum_{y \in N(x)} \frac{1}{M} \min\left(1, \frac{\pi_y}{\pi_x}\right)\right) \\ &= \sum_{y \in N(x), \pi_y \geq \pi_x} \pi_y \frac{1}{M} \cdot 1 + \sum_{y \in N(x), \pi_y < \pi_x} \pi_y \frac{1}{M} \frac{\pi_y}{\pi_x} + \pi_x \\ &- \sum_{y \in N(x), \pi_y \geq \pi_x} \pi_x \frac{1}{M} \frac{\pi_y}{\pi_x} - \sum_{y \in N(x), \pi_y < \pi_x} \frac{1}{M} \cdot 1 \\ &= \pi_x.\end{aligned}$$

Example.

$\Omega = \mathbb{Z} \cap [-1000{,}1000]$

$\pi \sim e^{-\frac{(x-\mu)^2}{2\delta}}$

```
X_0 arbitrary
for i = in range(1,m):
  y <- uniformly from {X_i - 11,X_i + 1}
  M <- uniformly from [0,1]
  if M ≤ π(y)/π(x):
    X_{i+1} = y
  else:
    X_{i+1} = X_i
return X_m
```

Example.

Find maximum of a positive function f.

Use metropolis algorithm to sample proportional to f.

Note: all I need to know is ratios $\frac{f(y)}{f(x)}$.

Back to independent sets.

$G = (V, E)$

$\Omega =$ independent sets.

$\lambda \in (1, \infty)$

$\pi(u) \sim \lambda^{|u|}$

$\pi(u) = \frac{\lambda^{|u|}}{\sum_{v \text{ independent set}} \lambda^{|v|}}$.

How to calculate the sum?

No problem: only need proportions.

X_0: arbitrary independent set.

$X_i \to X_{i+1}$:

- we pick $v \in V$ uniformly at random,
- if $v \in X_i \implies$
  - $X_{i+1} = X_i \setminus \{v\}$ qith probability $\frac{1}{\lambda} = \min\{1, \frac{\pi_y}{\pi_x}\}$,
  - $X_{i+1} = X_i$ with probability $1 - \frac{1}{\lambda}$,
- if $v \in X_j$ and $X_i \cup \{v\}$ is independent $\implies X_{i+1} = X_i \cup \{v\}$,
- otherwise $X_{i+1} = X_i$.

Example.

Bayes: $P(A \mid B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}$.

$B \leftarrow$ machine is giving values, e.g. $y_1 = 0.05, y_2 = -0.1, y_3 = 0.07, y_4 = 3$.

We believe $B \sim N(\mu, 0{,}05)$.

$\mu = laplacian(0, 0.01)$.

$P(\mu \mid B) = \frac{e^{\frac{|\mu|}{0.01}} e^{-\sum \frac{(x_i - \mu)^2}{0.05}}}{\int \cdots}$.

Integral is difficult to calculate.

Sample μ with Metropolis algorithm.

8.4 M.C. for 1-factor in bipartite graphs

G regular graph
$|A| = |B|$.
How to find 1-factor?
Augmenting paths.
Let M be (suboptimal) matching.
If we find $s - t$ path, we switch edges and get bigger matching.
Starting point.
G d-regular graph.
Graph $G = (A \cup B, E)$, M suboptimal matching.

- Add s and add directed edges to vertices in A that are not matched with weight d,
- add t and add directed edges to vertices in B that are not matched with weight d,
- orient edges in M from B to A that weight $d - 1$,
- orient edges in $E \setminus M$ from B to A that weight 1,
- we add edge from t to s that weight $(|A| - |M|)d$.

Observation:

- for each vertex $x :\ deg^-(x) = deg^+(x)$ (out weights = in weights),
- if $|A| > |M|$, then graph is eulerian $\implies$ there is an augmenting path.

How to find $s - t$ path?
Do a random walk.
Expected time to get from s to t is $h_{s,t}$
$\frac{1}{\pi(s)} = h_{s,s} = h_{s,t} + 1$.

Lemma 8.4.1.
Let X be a M.C. defined as a random walk on directed (weighted) graph with $deg^-(x) = deg^+(x)$ for each x. Then the stationary distribution is

$\pi = \left[\frac{deg^+(x_i)}{|E|}\right]_{i=1}^n$.

w_{ij}: weight from i to j.

Proof 8.4.2.
$\pi P = \pi \left[\frac{w_{ij}}{deg^+(x_i)}\right]_{i,j=1}^n = \left[\frac{\sum_j w_j i}{|E|}\right]_{i=1}^n = \left[\frac{deg^-(x_i)}{|E|}\right]_{i=1}^n = \left[\frac{deg^+(x_i)}{|E|}\right]_{i=1}^n$. ■

$h_{s,s} = \frac{1}{\pi_s} \leq \frac{|E|}{deg^+(s)} \leq \frac{3(|A|-|M|)d+|M|(d-1)+(|A|-|M|)d+|M|(d-1)}{(|A|-|M|)d} \leq \frac{4|A|}{|A|-|M|}$.
Expected time to find augmenting path $\leq \frac{4|A|}{|A|-|M|}$.
$|A| = n$
Expected time to find 1-factor $\leq \sum_{i=1}^{n-1} \frac{4n}{n-i} = 4n \sum_{i=1}^{n-1} \frac{1}{i} \leq 4n(1 + \ln n)$ - in $O(n \log n)$.

Network centrality

Degree as measure - natural idea.
Use M.C: walk randomly on the network, those that are visited more oftenly are more important.
Pagerank.
Let A be the adjacency matrix of G.
$P_{ij} = \alpha \frac{A_{ij}}{degi} + (1-\alpha)\frac{1}{n}$;
α: normal random walk,
$1-\alpha$: jump to any.
$\alpha = 0.85$.

Poglavje 9

Randomized incremented constructions (RIC)

Observation:

Let S be a set of n distinct elements.

Let $X_1 \dots X_n$ be a random permutation of the elements.

Let $S_i = \{X_1 \dots X_i\}$.

$P(X_i = \min(S_i)) = \frac{1}{i}$.

$Y = |\{j \in \{1 \dots n\} \mid j = \text{ minimal of } S_j\}|$

$Y = Y_1 + \dots + Y_n$

$$Y_j = \begin{cases} 1: & \text{if } i = \min S_i \\ 0: & \text{otherwise} \end{cases}$$

$E(Y) = \sum_{i=1}^{n} E(Y_i) = \sum_{i=1}^{n} \frac{1}{i}$ in $O(\log n)$.

```
Alg():
  X_1 ... X_n = random permutation of S
  min = X1
  for i in range(1, n+1):
    if Xi < min:
      print(\sn{HA})
      min = Xi
```

We get $O(\log n)$ „HA“ printed.

Incremental construction (IC).

Input $S = \{s_1 \dots s_n\}$.

We will build structures $DS(S_i)$:

$DS(S_1 \to \dots \to DS(S_n))$.

$DS(S_n)$ will help us give answer.

Randomized: permute S at the beginning.

9.1 Quicksort as RIC

S: set of elements we want to order.

$X_1 \dots X_n$: random permutation of S.

$S_i = \{X_1 \dots X_i\}$.

S_i splits $\mathbb{R}$.

Define $DS(S_i)$:

- save intervals: each interval will be saved by endpoints,
- for each interval we will be saving its points,
- for each X_j, $j > i$ we will save in which interval it is,
- for each left point of the interval we will save the right point.

```
QuicksortRIC(S):
  # start of DS(Si)
  I = [(-∞,∞)]
  P[(-∞,∞)] = S
  for each X_i:
    Int(X_i) = (-∞,∞)
  Next(-∞) = ∞
  # end of DS(Si)
  for i in range(1,n+1):
    I_i = Int(X_i) = (X_j,X_k) # Ii splits interval (X_j,X_k)
```

$I_{i1} = (X_j, X_i)$
$I_{i2} = (X_i, X_k)$
for $Xl \neq X_i, X_l \in P(I)$:
  add X_l **to** $P(I_{i1})$ **or** $P(I_{i2})$ **depending on** $X_l < X_i$ **or** $X_l > X_i$
$Next(X_j) = X_i$
$Next(X_i) = X_k$
return $[Next(-\infty), Next(Next(-\infty)) \dots]$

Similarity to quicksort: spliting intervals.

Analysis:
for set i, we need $O(|P(I_i)|)$,
$E(|P(I_i)|) = ?$
e.g.
if $x_4 = a_4$:
if $x_4 = a_2$:
$P(X_i = a_j) = \frac{1}{i},\ j \in \{1, 2 \dots i\}$.
Expected value of steps in iteration i
$\sum_{j=1}^{i} \frac{1}{i} \left(P\left((a_{j-1}, a_j)\right) + P\left((a_j, a_{j+1})\right)\right) \leq \frac{1}{i} 2(n-i) \leq \frac{2n}{i}$

$$
\begin{aligned}
E\,(\text{number of steps in QuicksortRIC}) &\leq \sum_{i=1}^{n} \frac{2n}{i} \\
&\leq 2n(1 + \log n) \quad \to \text{ in } O(n \log n).
\end{aligned}
$$

9.2 Linear programming

Task: maximize $f(x_1 \dots x_n) = c_1 x_1 + \cdots + c_d x_d$.
Constraints:

$$\begin{aligned}
&a_{11}x_1 + \cdots + a_{1d}x_d \leq b_1 \\
&\vdots \\
&a_{n1}x_1 + \cdots + a_{nd}x_d \leq b_n.
\end{aligned}$$

Geometric interpretation.
Cases:

- infeasible region
- unbounded
- multiple solutions.

Alg:

- symplex algorithm worst case $O\left(2^n\right)$,
- interior point method (polynomial algorithm).

Seidel's algorithm:
running in expected $O(n)$ time when d is constant.
One dimension.

$$\begin{aligned}
\max\ & cx \\
& a_1 x \leq b_1 \\
& \vdots \\
& a_n x \leq b_n,
\end{aligned}$$

where n is number of constraints.

- a_i positive: $(-\infty, \frac{b_i}{a_i}]$ $\left(x_i \leq \frac{b_i}{a_i}\right)$,
- a_i negative: $[\frac{b_i}{a_i}, \infty)$ $\left(x_i \geq \frac{b_i}{a_i}\right)$.

$a_i \neq 0$.
Alg:

$R = \min_i\{\frac{b_i}{a_i}; a_i > 0\}$,

$L = \max_i\{\frac{b_i}{a_i}; a_i < 0\}$,

if $L > R$: program infeasible,

else:

if $c > 0$: return R,

if $c < 0$: return L.

2-dim: assume general position.

$$\begin{aligned}\max\ & c_1x + c_2y \\ & a_{11}x + a_{12}y \leq b_1 \\ & \vdots \\ & a_{n1}x + a_{n2}y \leq b_n \\ & x \leq M \text{ or } x \geq -M \\ & y \leq M \text{ or } y \geq -M.\end{aligned}$$

$\leq, \geq$ depending on c_1, c_2.
Notation:

h_i: halfspace defined by $a_{i1}x + a_{i2}y \leq b_i$,

m_i: added halfspaces, defined by $X, Y \leq M$ or $\geq -M$,

l_i: line that bounds.

Alg:

- first randomly permute h_i,

- $H_i = \{m_1, m_2, h_1 \dots h_i\}$,
- $v_i \in \cap H_i$ optimal solution after i constraints,
- $v_0 = (\pm M, \pm M)$,
- inductively add h_i.

Cases:

if $v_{i-1} \in h_i \implies v_i = v_{i-1}$,

if $v_{i-1} \notin h_i \implies v_i \in h_i$:

$a_{i1}x + a_{i2}y = b_i$

a_{i1} or $a_{i2} \neq 0$, e.g. a_{i1};

$x = \frac{b_i - a_{i2}y}{a_{i1}}$.

Insert x in all constraints $\implies$ linear program in 1-dim, i (i-1?) constraints $\implies$ get v_i in $O(i)$.
Analysis:

- worst case: $\sum_{i=1}^n O(i) = O(n^2)$,
- expected: $E(X) = \sum_{i=1}^n E(X_i)$,
- $X_i =$ running time of i-th iteration,
- $X_i = \begin{cases} O(1): & \text{case 1} \\ O(i): & \text{case 2} \end{cases}$
- $P(\text{case 2}) \leq \frac{2}{i}$ - optimal point on at most 2 lines,
- $E(X) \leq \sum_{i=1}^n O(1) \cdot 1 + O(i) \cdot \frac{2}{i} = O(n)$.

d-dim

- constraints define half-spaces,
- boundary is hyperplane ($d-1$ dimensional),

- general position: intersection of $d-i$ hyperplanes is i dimensional, intersection of $d+1$ hyperplanes is $\emptyset$.

Alg:

first add $X_i \leq M$ or $X_i \geq -M$ depemding on c_i,

random permutation $(h_1 \ldots h_n)$,

$H_i = \{m_1 \ldots m_d, h_1 \ldots h_i\}$,

$v_0 \in \cap \partial m_i$,

inductively add h_i:

$v_{i-1} \in h_i \implies v_i = v_{i-1}$,

$v_{i-1} \notin h_i \implies$ we need to solve LP in $d-1$ dimensions with i constraints ($O(i)$ expected),

$P(v_{i-1} \notin h_i) \leq \frac{d}{i}$,

$E(X) \leq \sum_{i=1}^{n} O(1) + \frac{d}{i} O(i) = O(n)$.

X: running time.
Careful implementation runs in $O(d!\, n) \implies$ very useful for low dimensions.
Problem: let P be convex polygon given by ordered set of vertices
$y = a_i x + b_i$.
Find largest disc embeddable in P.
Input: $P_1 \ldots P_n$,

output: $(s_1, s_2), r$.

$$\begin{aligned} &\max r \\ &\qquad r = \left| \frac{s_2 - a_i s_1 - b_i}{\sqrt{a_i^2+1}} \right| \\ &\qquad \frac{s_2 - a_i s_1 - b_i}{\sqrt{a_i^2+1}} \geq r \text{ - line above } P \\ &\qquad -\frac{s_2 - a_i s_1 - b_i}{\sqrt{a_i^2+1}} \leq -r \text{ - line below } P \end{aligned}$$

$\implies$ LP in 3 dim.

Note: $\frac{s_2 - a_i s_1 - b_i}{\sqrt{a_i^2+1}}$ positive if (s_1, s_2) above the line, negative otherwise.

Poglavje 10

Hashing

A hash function is a randon function,
$h : U \to \{0, 1 \ldots n-1\} = M$,
U - universe,
$u = |U|$,
$m = |M|$.
Ideally we would like for h to be as completely random: $P(h(x) = t) = \frac{1}{m}$.
Standard application.
Let $V \subset U$, $|V| << |U|$.
We would like to quickly answer if $x \in V$ for every $x \in U$.
Solution:

- take $h : U \to M$,
- make a table $T = [0, 1 \ldots n-1]$,
- for $v \in V$:

  $T[h(v)] = 1$,

  $T[y] = 0 \; \forall y \in h(V)$.

- Let $x \in V$. Check

- if $T[h(x)] = 1 : \; x \in V$,
- else: $x \notin V$.

Note: this is not OK: h not injective.

For $x \in U$, tell if $x \in V$ in $O(1)$.
$h = \text{SHA256} : U \to \{0,1\}^{256}$.
Approach:

- design a family of hash functions,
- study collisions $P_h(h(x) = h(y))$,
- H meeds to be „simple“.

Bad example: H = all functions from U to M storing $h \in H$ would take $|U| \log_2 |M|$ bits.

Definition 10.0.1. A family of hash functions to be universal if for
$\forall x, y \in U, x \neq y, h \in H : \; P(h(x) = h(y)) \leq \frac{1}{m}$ (probability of collision).

k-independent if $\forall x_1 \dots x_k \in U$ pairwise different, $\forall t_1 \dots t_k \in M$
$P_r(h(x_i) = t_i \; \forall i) \leq \frac{1}{m^k}$.

Example.
$U = \{0, 1, 2, 3\}$,
$M = \{0, 1\}$,
$H = \{h_0, h_1, h_2\}$,
$h_0 : \{0 \to 0, 1 \to 0, 2 \to 1, 3 \to 1\}$,
$h_1 : \{0 \to 0, 1 \to 1, 2 \to 0, 3 \to 1\}$,
$h_2 : \{0 \to 0, 1 \to 1, 2 \to 1, 3 \to 0\}$.
$P(h(0) = h(2)) = \frac{2}{3} > \frac{1}{2}$ - not universal.

Why universal?
U, V, M
H universal: $\forall x, y : \; P(h(x) = h(y)) \leq \frac{1}{m}$.
X: number of collisions of V.

$E(X) = E\left(\sum_{x,y \in V, x \neq y} X_{x,y}\right)$

$X_{x,y} = \begin{cases} 1: & \text{if } h(x) = h(y) \\ 0: & \text{else} \end{cases}$

$E(X) = \sum_{x,y \in V, x \neq y} E(X_{x,y}) \leq \binom{n}{2} \cdot \frac{1}{m}.$

U, V, M, H

$T[0 \ldots m-1]$

$\forall v \in V$

$T[h(v)] = v.$

For $x \in V$ we check $T[h(x)]$ if equals x,

for $y \in U \setminus V$, $T[h(y)] \neq y$.

For $z \in V$, $T[h(z)]$ can happen $\neq z$ if h has collisions in V.

Lemma 10.0.2.

Let $m \geq n^2$ and H universal. Then the probability that h has no collisions in $V \ \geq \frac{1}{2}$.

Proof 10.0.3.

X: number of collisions

$E(X) \leq \binom{n}{2} \cdot \frac{1}{m} < \frac{n^2}{2} \cdot \frac{1}{n^2} = \frac{1}{2}$

$P(X \geq 1) \overset{\text{Markov}}{\leq} \frac{E(X)}{1} = \frac{1}{2}$

$P(X = 0) \geq \frac{1}{2}.$ ■

Example (Universal hash family).

$U = \{0, 1 \ldots u-1\}$ (bits $\equiv$ numbers)

$M = \{0, 1 \ldots m-1\}$.

Define: let $p \geq u$, p prime number.

Define for $a, b \in \mathbb{Z}_p,\ a \neq 0$.

$h_{a,b} = (ax + b) \mod m$

$ax + b \in \mathbb{Z}_p$

$H = \{h_{a,b} \mid a, b \in \mathbb{Z}_p,\ a \neq 0\}.$

Proof 10.0.4.

$P(h_{a,b}(x) = h_{a,b}(y)) =?$

x, y fixed.

For any a, b denote

$ax + b = t_x$

$ay + b = t_y$:

$a \sqcup +b \in \mathbb{Z}_p$.

$$\begin{bmatrix} x & 1 \\ y & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$\det \begin{bmatrix} x & 1 \\ y & 1 \end{bmatrix} \neq 0$, because $x \neq y$

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} x & 1 \\ y & 1 \end{bmatrix}^{-1} \begin{bmatrix} t_x \\ t_y \end{bmatrix}.$$

For each t_x, t_y there exists 1 a, b mapping to t_x, t_y.

$h_{a,b}(x) = h_{a,b}(y) \iff t_x = t_y \mod m$.

This holds for $p\left(\lceil \frac{p}{m} \rceil + 1\right)$

p: choice of t_y

$t_x = t_y + km$

$P\left(h_{a,b}(x) = h_{a,b}(y)\right) \leq \frac{p\left(\lceil \frac{p}{m} \rceil - 1\right)}{p(p-1)} \leq \frac{\frac{p-1}{m}}{p-1} = \frac{1}{m}$. ■

Function random for 2 elements, fixed for ≥ 3.

Higher k-independent: better.

10.1 Chaining

$V, U, h : U \to V$.

Answer $x \in V$ in $O(1)$.

$T[0 \ldots m-1]$

$n = |V|$

$\forall v \in V$:

$h(v_1) = h(v_2) \rightarrow [v_1\ v_2 \ldots]$ - linked list.

Now:

$x \in U$.

Check if x is in list at $T[h(x)]$.

Check takes $O(\text{length of a list at } h(x)) = 1+$ number of collisions with x.
X_x: number of collisions with x.
$E(X_x) = \sum_{y \in V} E(X_{x,y}) \leq n \cdot \frac{1}{m}$ if hash function is universal.
$\alpha = \frac{n}{m}$: load factory (how many elements in 1 place).
$E(X_x) = 1$
$E(\max_x X_x) \neq \max_x E(X_x) = 1$.

Theorem 10.1.1. Assume we throw n balls into n bins uniformly at random. Then with high probability the fullest contains $\theta\left(\frac{\log n}{\log(\log n)}\right)$ balls.

Proof 10.1.2.
$\stackrel{?}{\leq} \frac{3 \ln n}{\ln \ln n}$.
Let X_j be the number of balls in bin j.
$P\left(X_j \geq \frac{3 \ln n}{\ln \ln n}\right) = P(\text{there exists subset } S \text{ of balls thrown to bin } j)$.
$|S| = k$

$$
\begin{aligned}
& P\left(\cup_{S \text{ balls},|S|=k} \text{ balls from } S \text{ are thrown to bin } j\right) \\
& \leq \sum_{S \text{ balls},|S|=k} P(\text{balls from } S \text{ are thrown to } j) \\
& = \binom{n}{k}\left(\frac{1}{n}\right)^k \\
& \leq \frac{n^k}{k!} \cdot \frac{1}{n^k} \\
& = \frac{1}{k!} & \stackrel{10.1}{\leq} \frac{e^k}{k^k} \\
& = \left(\frac{e \ln n}{3 \ln \ln n}\right)^{\frac{3 \ln n}{\ln \ln n}} \\
& \leq e^{\frac{3 \ln n}{\ln \ln n} \cdot (\ln \ln \ln n - \ln \ln n)} \\
& = e^{-3 \ln n + \frac{\ln \ln \ln n \cdot (\ln n \cdot 3)}{\ln \ln n}} & \stackrel{10.2}{\leq} e^{-3 \ln n + \ln n} \\
& = \frac{1}{n^2};
\end{aligned}
$$

$$e^x = \sum_{i=1}^{\infty} \frac{k^i}{i!} \geq \frac{k^k}{k!}, \tag{10.1}$$

$$\frac{\ln \ln \ln n}{\ln \ln n} \to 0. \tag{10.2}$$

$P(\text{at least for 1 bin } j \geq k) = n \cdot \frac{1}{n^2} = \frac{1}{n}$. ■

U, V, H hash family, $h : U \to M$
$v \in V$
$n = |V|$
max load $O\left(\frac{\log n}{\log(\log n)}\right)$.
Perfect hashing: we would like

- $O(1)$ lookup (worst case)
- $O(n)$ size of table.

10.2 2 level hashing

Input: V
$n = |V|$.
Take hash function from universal family with $m = |M| = n$.
Count total collisions X.
$E(X) \leq \binom{n}{2} \cdot \frac{1}{m} \leq \frac{n}{2}$
$P(x \geq n) \overset{\text{Markov}}{\leq} \frac{1}{2}$
$\implies$ by repeating sample h we can guarantee

- for each $i \in M$ we store at $T[i]$ another hash table of size C_i^2, where C_i = number of elements of V, hashed in i,
- we sample h_i from universal hash family with $M_i = C_i^2$.

$P(h_i \text{ has no collisions}) \geq \frac{1}{2}$ (by lemma).
We resample if h_i has collisions.
$E(\text{sampling } h_i) = 2$.
Construction time:

- step 1: $O(n)$
- step 2: $O(C_1 + \cdots + C_n) = O(n)$;

together $O(n)$.
Lookup time: $O(1)$ (evaluating $h(x)$ and $h_{h(x)}(x)$).
Space: $O(C_1^2 + \cdots + C_n^2)$ in $O(n)$.
By first step:
$n >$ number of collisions of $h = \sum_{i=1}^{n} \binom{C_i}{2} = \sum_{i=1}^{n} \frac{C_i^2 - C_i}{2}$
$\implies \sum_{i=1}^{n} C_i^2 < 2n + \sum_{i=1}^{n} C_i = 3n$.

10.3 The power of 2 choices

Variant: placing n balls in n bins but for each ball we choose d bins uniformly at random and put the ball in bin with minimal load.

Theorem 10.3.1. The above process with $d \geq 2$ results in at most maximum load of $O\left(\frac{\ln(\ln n)}{\ln d}\right)$.

Proof 10.3.2. (sketch).
b_i = upper bound of the number of bins with load at most i.
Height of a ball = the number of balls in the bin, where the ball is placed.
$P(\text{a ball has height at least } i+1) \leq \left(\frac{b_i}{n}\right)^d$ (choose d times independently).
X^{i+1}: number of balls with height $\geq i+1$.
$X^{i+1} = \sum_{j=1}^{n} X_j^{i+1}$
X_j^{i+1}: indicator variable of j-th ball having height $i+1$.
$E(X^{i+1}) \leq \sum_{j=1}^{n} \left(\frac{b_i}{n}\right)^d = n \cdot \left(\frac{b_i}{n}\right)^d$.
Chernoff bound: with high probability $X^{i+1} \leq 2n \left(\frac{b_i}{n}\right)^d$.
$X^{i+1} \geq$ number of bins with load at least $i+1$.
Define (set)
$b_{i+1} = \frac{\sum b_i^d}{n^{d-1}}$
$b_4 = \frac{n}{4}$
$b_{i+4} \stackrel{?}{=} \frac{n}{2^{2 \cdot d^i - \sum_{j=0}^{i-1} d^j}}$

$i = 0$: $b_4 = \frac{n}{2^{2^1}} = \frac{n}{4}$

$i \to i+1$:

$$\begin{aligned} b_{i+4} &= \frac{2 \cdot b_{i+3}}{n^{d-1}} \\ &\stackrel{IH}{=} \frac{2 \cdot \left(\frac{n}{2^{2 \cdot d^i - \sum_{j=0}^{i-1} d^j}} \right)^d}{n^{d-1}} \\ &= \frac{2^1 \cdot n^d}{n^{d-1} \cdot 2^{2 \cdot d^{i+1} - \sum_{j=1}^{i} d^j}} \\ &= \frac{n}{2^{2 \cdot d^{i+1} - \sum_{j=0}^{i} d^j}}. \end{aligned}$$

In particular: $b_{i+4} \leq \frac{n}{2^{d^i}} < 1$ when?

$$\begin{aligned} n &< 2^{d^i} \\ \log_2 n &< d^i \\ \log_d \log_2 n &< i \end{aligned}$$

$\implies$ for $i = \frac{\log(\log_2 n)}{\log d}$ is $b_i < 1 \implies$ no bins with load $> \frac{\log(\log_2 n)}{\log d}$. ■

Application:
We sample 2 hash functions $h_1, h_2 : U \to M$.
For element $v \in V$ we insert in $T[h_1(v)]$ or $T[h_2(v)]$ depending on which list is shorter.
Max load in $O(\log(\log n))$.

10.4 Cockoo hashing

Idea: use 2 hash functions but allow moving elements later.
We want to have at most 1 element at each entry in the table.
Inserting:

- if empty: insert,
- if not empty: push other element to its other choise, repeat recursively.

Questions:

- how many do I need to move,
- how many elements can I insert before problems?

We can think of positions in the table as vertices and elements of V as edges. $|V|$ edges are inserted uniformly at random (if ideal hash function) $\implies$ random graph.
Erdös-reny model: $G_{n,m} \approx G_{n,p}$ if $m = \binom{n}{2} \cdot p$ (A.S. properties).
If $np < 1 - \varepsilon$: all connected components have size at most $O(\log n)$, components are trees or at most 1 cycle per component, expected size of a component is $O(1)$.
Fact: if graph has at most 1 cycle per component, then inserting can be done and takes at most 2$\cdot$(size of component) time (each edge changes direction at most 2 times).

Theorem 10.4.1.
Let $n = |U|$, $h_1, h_2 : U \to M$, $m = |M| = 2 \cdot (1 + \varepsilon) \cdot n$, then with high probability cockoo hashing works correctly with

- inserting time:
  - $O(\log n)$ time worst case,
  - $O(1)$ expected case,
- space: $O(n)$,
- lookup time: $O(1)$.

Dynamically add element:
$m = 2 \cdot (1 + \varepsilon) \cdot n$
$p = \frac{m'}{\binom{n'}{2}} = \frac{2m'}{n'(n'-1)}$
$pn' = \frac{2m'}{(n'-1)} = \frac{2n'}{2(1+\varepsilon)n'} = \frac{1}{1+\varepsilon} < 1 + \varepsilon'$

10.5 Bloom filter

Take k hash functions $h_1 \dots h_k$ at random, $h_i : U \to M, T[0 \dots m-1]$.
$V \subset U$, for every element $v \in V$ set $T[h_i(v)] = 1 \; \forall i \in \{1 \dots k\}$.
False positives: $x \notin V$ such that $T[h_i(x)] = 1 \; \forall i \in \{1 \dots k\}$.
For each $T[j]$ $P(T[j] = 0) = \left(\left(1 - \frac{1}{m}\right)^n\right)^k \approx e^{-\frac{nk}{m}}$;
k: each hash function, n: for each v.
Now
$P\left(T[h_i(x)] = 1 \; \forall i, \forall x \notin V\right) \approx \left(1 - e^{-\frac{nk}{m}}\right)^k = f(k)$ - probability of a false positive.
$\left(1 - e^{-\frac{nk}{m}}\right)$: 1 position.
Searching for a minimum:
$f'(k) = 0$
$\implies k = \ln 2 \cdot \frac{m}{n}$
$f\left(\ln 2 \cdot \frac{m}{n}\right) = \left(\frac{1}{2}\right)^{\ln 2 \frac{m}{n}} \approx 0.6185^{\frac{m}{n}}$
$\implies$ we choose m such that $0.6185^{\frac{m}{n}}$ small (in $O(n)$)
$\implies$ calculating $k = \ln 2 \cdot \frac{m}{n}$

$\implies$ hashing with space $O(n)$

$\implies$ checking in $O(1)$

$\implies$ probability of error small.

10.6 Linear probing

$V \subset U$, $h : U \to M$, $T[0 \dots m-1]$.

- Insert $v \in V$: check $T[h(v)], T[h(v)+1], T[h(v)+1] \dots$ until finding empty space, then insert it.
- Check if $x \in V$ by checking $T[h(x)] \stackrel{?}{=} x, T[h(x)+1] \stackrel{?}{=} x \dots$ until finding x or finding empty.

$x \in U$

X: number of steps to check if $x \in V$.

$E(X) =?$

Block of size 2^l is bad if it has more than $2^l \cdot \frac{2}{3}$ values.

Set $\frac{n}{m} = \frac{1}{3}$.

Expected number of elements hashed in block of size 2^l is $\frac{1}{3} \cdot 2^l$.

$$\begin{aligned} E(X) &= \sum_{i=0}^{n} P(X = i) \cdot i \\ &\leq \sum_{j=0}^{\log_2 n} P(2^{j-1} < X \leq 2^j) \cdot 2^j \\ &\leq \sum_{j=0}^{\log_2 n} P(\text{block above } h(x) \text{ of size } 2^j \text{ is bad}) \cdot c \cdot 2^j. \end{aligned}$$

c: not aligned?

$P(\text{block of size } 2^j \text{ is bad}) = P(Y > \frac{2}{3} \cdot 2^j) = P(Y - \frac{1}{3} \cdot 2^j > \frac{1}{3} \cdot 2^j)$;

Y: number of elements hashed to the block.

$E(Y) = \frac{1}{3} \cdot 2^j$

$E(X) \overset{\text{Chernoff}}{\leq} e^{-k \cdot 2^j}$; Chernoff: sum of independent indicators.

$E(X) < O(1) \cdot \sum_{j=0}^{\log_2 n} 2^j \cdot e^{-k \cdot 2^j}$ in $O(1)$

$\implies$ checking in $O(1)$.

Chernoff: if ideal hash function; 5 independent is enough.

Poglavje 11

Data streams

Stream of values
$\sigma = a_1, a_2 \dots a_n$
a_i: tokens
$a_i \in [n]$
m: length of stream (very large).

Definition 11.0.1. $f_i = |\{j \mid a_j = i\}|$

We could be interested in

- number of different token,
- frequency of some token,
- frequent tokens: $\{i \in [n] \mid f_i \geq \frac{m}{10}\}$
- moments: $\|f\|^2 = \sum_{i \in [n]} f_i^2$
- $\vdots$

We want to use memory in $O(poly(\log n, \log m)) << O(n, m)$.
Most problems cannot be solved precisely, hence we search for (ε, δ)-approximation.
Algorithm $A(G)$:

- initialitazion,

- incremental steps,
- finalization

using randomness (oblivious stream - it doesn't know which randomly, e.g. we can choose stream that „attacks algorithm").

11.1 Count min sketch

For a given $i \in [n]$ (token) at the end of stream give f_i.

```
A(σ, ε, δ):
  Init: k = ⌈2/ε⌉, t = ⌈log_2(1/δ)⌉.
  We choose t hash functions h_1 ... h_t : [n] → M = [k] = {1 ... k}
  from a universal family H.
  Let C[0 ... t-1][0 ... k-1] be 2-dim (hash) table
  C[i][j] = 0 ∀i,j.
  Updates:
      for every token a_i ∈ σ we update C
        for j = 0,1 ... t-1:
          C[i][h_i(a_j)] += 1
  Output: we asked a ∈ [n] return f̄_a = min_{0≤j≤t-1} C[j][h_j(a)];
    min collisions.
```

Theorem 11.1.1.
For every $a \in [n]$ it holds
$f_a \leq \overline{f_a} \leq f_a + \varepsilon m$
with probability at least $1 - \delta$.

Notice: space needed $O(t \cdot k \cdot \log m) = O\left(\frac{2}{\varepsilon} \cdot \log_2\left(\frac{1}{\delta}\right) \log m\right)$.

Proof 11.1.2.
$\forall i \in [t] : C[i][h_i(a)] \geq f_a \implies \overline{f_a} \geq f_a$.
Fix a.
Let $X_i = C[i][h_i(a)] - f_a$ excess of i-th count.

$I^i_{x,y} = \begin{cases} 1: & \text{if } h_i(x) = h_i(y) \\ 0: & \text{else} \end{cases}$
$X_i = \sum_{y\in[n],y\neq a} I^i_{x,y} \cdot f_y$.

$$\begin{aligned} E(X_i) &= \sum_{y\in[n],y\neq a} E(I^i_{x,y}) \cdot f_y \\ &\overset{11.1}{\leq} \sum_{y\in[n],y\neq a} \frac{1}{k} \cdot f_y \\ &\leq \frac{1}{n} \cdot m \\ &\overset{11.2}{\leq} \frac{m}{2}; \end{aligned}$$

$$\text{hash function from universal family,} \tag{11.1}$$

$$P(X_i \geq \varepsilon m) \overset{\text{Markov}}{\leq} \frac{\varepsilon m}{2\varepsilon m} = \frac{1}{2} \text{ for fixed } i. \tag{11.2}$$

$$\begin{aligned} P(\overline{f_a} - f_a \geq \varepsilon m) &\leq P(X_i \geq \varepsilon m\ \forall i) \\ &\overset{\text{indep.}}{=} \left(\frac{1}{2}\right)^t \leq \delta. \end{aligned}$$

■

11.2 Estimating the number of distinct elements

We want $d = |\{i \in [n], f(i) > 0\}|$.
Define for $x \in \mathbb{N}$:
$zeros(x) = \max\{i \mid 2^i \text{ divides } x\}$: number of zeros at the end in binary representation of x.

```
Alg(σ):
  Init:
    h: random hash function from 2-independent family.
    # recall: [n]: all possible elements of σ.
```

```
    h : [n] → [n]
    unlog? n = 2^{n'}
    z = 0
  Update:
    a_i ∈ σ
    if zeros(h(a_i)) ≥ z:
      z = zeros(h(a_i))
  Output:
    d̄ = 2^{z+1/2}
```

Define $\forall a \in [n], r \in \mathbb{N}$

$X_{r,a} = \begin{cases} 1: & \text{if } zeros(h(a)) \geq r \\ 0: & \text{else} \end{cases}$

$Y_r = \sum_{a \in \sigma} X_{r,a}$.

Let $\overline{z}$ be z at the end of the algorithm: $\overline{d} = 2^{\overline{z}+\frac{1}{2}}$.

Notice:

$Y_r > 0 \iff \overline{z} \geq r$

$Y_r = 0 \iff \overline{z} < r$.

Lemma 11.2.1.

$P(X_{r,a} = 1) = \frac{1}{2^r}$,

$P(X_{r,a_1} = 1 \wedge X_{r,a_2} = 1) = \frac{1}{(2^r)^2}$.

Proof 11.2.2.

$P(X_{r,a} = 1) = P(zeros(h(a)) \geq r) = \frac{2^{n'-r}}{2^{n'}} = \frac{1}{2^r}$;

$2^{n'}$: all,

$2^{n'-r}$: fixed.

$P(X_{r,a_1} = 1 \wedge X_{r,a_2} = 1) \overset{h\ 2\ \text{indep.}}{=} P(X_{r,a_1} = 1) \cdot P(X_{r,a_2} = 1) = \frac{1}{(2^r)^2}$. ■

$P(\overline{d} \geq 3d)$ small?

$E(Y_r) = \sum_{a \in \sigma} E(X_{a,r}) = \sum_{a \in \sigma} \frac{1}{2^r} = \frac{d}{2^r}$

Let $k \in \mathbb{N}$ be such that $2^{k+\frac{1}{2}} \geq 3d > 2^{k-\frac{1}{2}}$.

$$
\begin{aligned}
P\left(\overline{d} > 3d\right) &\leq P\left(2^{\overline{z}-\frac{1}{2}} > 2^{k-\frac{1}{2}}\right) \\
&= P\left(\overline{z} + \frac{1}{2} > k - \frac{1}{2}\right) \\
&= P(\overline{z} \geq k) \\
&\overset{\text{lemma}}{=\!=} P(Y_k > 0) \\
&\overset{\in\mathbb{N}}{=\!=} P(Y_k \geq 1) \\
&\overset{\text{Markov}}{\leq} \frac{E(Y_k)}{1} = \frac{d}{2^k} \\
&\overset{k}{\leq} \frac{d \cdot 2^{\frac{1}{3}}}{3d} = \frac{\sqrt{2}}{3}.
\end{aligned}
$$

$P(\overline{d} \leq \frac{d}{3})$ small?

Let $l \in \mathbb{N}$ be such that $2^{l-\frac{1}{2}} \leq \frac{d}{3} < 2^{l+\frac{1}{2}}$.

$$
\begin{aligned}
P\left(\overline{d} < \frac{d}{3}\right) &\leq P\left(2^{\overline{z}+\frac{1}{2}} < 2^{l+\frac{1}{2}}\right) \\
&= P\left(\overline{z} + \frac{1}{2} < l + \frac{1}{2}\right) \\
&= P(\overline{z} \leq k) \\
&\overset{\text{lemma}}{=\!=} P(Y_l = 0) \\
&= P\left(Y_l - \frac{d}{2^l} < -\frac{d}{2^l}\right) \\
&\leq P\left(|Y_l - \frac{d}{2^l}| \geq \frac{d}{2^l}\right) \\
&\overset{\text{Chebisev}}{\leq} \frac{Var(Y_l)}{\left(\frac{d}{2^l}\right)^2} \\
&\overset{l}{\leq} \frac{d \cdot 2^{\frac{1}{3}}}{3d} = \frac{\sqrt{2}}{3};
\end{aligned}
$$

$$\begin{aligned}
Var(Y_l) &= Var\left(\sum_{a\in\sigma} X_{a,l}\right) \\
&\overset{h \text{ 2-indep.}}{=\!=\!=} \sum_{a\in\sigma} Var(X_{a,l}) \\
&= \sum_{a\in\sigma} E(X_{a,l}^2) - E(X_{a,l})^2 \\
&\overset{E(X_{a,l})\in\{0,1\}}{\leq} \sum_{a\in\sigma} E(X_{a,l}) \\
&= \frac{d}{2^l}.
\end{aligned}$$

$P\left(\frac{d}{3} < \overline{d} < 3d\right) \geq 1 - \frac{2\sqrt{3}}{3}$.
We use algorithm k-times, getting $\overline{d_1} \ldots \overline{d_k}$ (we need independent hash functions).
Define: $\overline{d} = median(\overline{d_1} \ldots \overline{d_k})$.

$$\begin{aligned}
P(\overline{d} \geq 3d) &= P\left(\text{at least } \left\lceil \frac{k}{2} \right\rceil \overline{d} - s \text{ are } \geq 3d\right) \\
&= P\left(X \geq \frac{k}{2}\right) \leq e^{-ck};
\end{aligned}$$

c: some constant,
$X = \sum_{i=1}^{k} X_i$,
$X_i = \begin{cases} 1: & \text{if } \overline{d_i} \geq 3d \\ 0: & \text{else} \end{cases}$
$P\left(\overline{d} \leq \frac{d}{3}\right) = \ldots$

Poglavje 12

Interactive proofs

A protocol between P prover and V verifier for function f.
Both share x,
r: randomness used,
P, V: algorithms,

$$out(V, x, r, P) = \begin{cases} 1: & V \text{ agrees that } f(x) = y \\ 0: & \text{else} \end{cases}.$$

Goal: minimal communication, minimal work for V.
Completeness:

- for every $x \in D$ (domain)
- $P(out(V, x, r, P) = 1) \geq 1 - \delta_c$ for some $\delta_c \in [0, 1)$.

Soundness:

- for every x such that $f(x) \neq y$
- $P(out(V, x, r, P^{'}) = 1) \leq \delta_s$ for every $P^{'}, \delta_s \in [0, 1)$.

Computational soundness:

- soundness,
- $P^{'}$ computationally bounded.

Zero-knowledge:

- informally: verifier learns nothing behind the claim.

Example.
Input: G graph,
$f(G) = \begin{cases} 1: & \text{if } G \text{ hamiltonian} \\ 0: & \text{else} \end{cases}$
$G \to P \overset{m_1:(v_1 \ldots v_n)}{\longrightarrow} V \leftarrow G$,
V: verifies that m_1 is hamiltonian cycle.
Proof: $O(n)$.
Verifier com. $O(n)$.

Example.
Input: A, B matrices,
$f(A, B) = A \cdot B$,
$(A, B) \to P \overset{C}{\to} P \leftarrow (A, B)$.
P: compute $C = A \cdot B$, send C,
V: check $A(Bv_i) = Cv_i$ for random v_i.
Prover: matrix multiplication $O(n^3)$ $(O(n^{\log_2(7)}))$.
Verifier: $O(n^2)$.
Proof size: $O(n^3)$ (possible to reduce is $O(\log n)$).

Example.
Input: $(n, y) \in \mathbb{N}^2$,
$f(n, y) = \begin{cases} 1: & \text{if there exists } x \text{ such that } y = x^2 (\text{mod } n) \\ 0: & \text{else} \end{cases}$;
qoaroatic?? reducibility problem.
$(n, y) \to P \to V \leftarrow (n, y)$.
P: sample $r \in \mathbb{Z}_n$, $s = r^2$, send s,
V: sample $b \in \{0, 1\}$, send b,
P: if $b = 0$: $m_2 = r$, if $b = 1$: $m_2 = r \cdot x$, send m_2,
V: accepts if $m_2^2 = s \cdot y^b$.

Completeness:

$$m_2^2 \stackrel{?}{=} s \cdot y^b$$

if $b = 0$:

$$m_2 = r$$

$$r^2 = m_2^2 = s \checkmark$$

if $b = 1$:

$$m_2^2 = sy$$

$$r^2x^2 = sy \checkmark (r^2 = s, x^2 = y)$$

Soundness:

- 2 options for what prover does.
  - Send s such that there is no r that $r^2 = s$.
    Then with probability $\frac{1}{2}$ is $b = 0$.
    Then prover needs to send m_2 such that $m_2^2 = s$ (impossible)
    $\implies$ fail with probability at least $\frac{1}{2}$.
  - Send s such that $r^2 = s$.
    Then with probability $\frac{1}{2}$ is $b = 1$.
    $m_2^2 = sy = r^2y \implies y = (m_2r^{-1})^2 \implies \exists x : x^2 = y$: contradiction
    $\implies$ fail with probability at least $\frac{1}{2}$.

With zero-knowledge.

12.1 Sum-check protocol

Let $g(x_1 \dots x_n)$ be multivariate polynomial of degree d over $\mathbb{F}$.
Let $H_g = \sum_{b_1 \dots b_n \in \{0,1\}} g(b_1 \dots b_n)$.
P wants to convince V that $c = H_g$.
$g \to P \to V \leftarrow g$.

P: sends c,
P: compute $g_1(x) = \sum_{b_2 \dots b_n \in \{0,1\}} g(x, b_2 \dots b_n)$, send $g_1(x)$,
V: check $g_1(0) + g_1(1) = c$, $deg(g) \leq d$, sample $r_1 \in \mathbb{F}$, send r_1,
for $j = 2 \dots n-1$:

P: compute $g_j(x) = \sum_{b_{j+1} \dots b_n \in \{0,1\}} g(r_1 \dots r_{j-1}, x, b_{j+1} \dots b_n)$, send $g_j(x)$,

V: checks $g_j(0) + g_j(1) = g_{j-1}(r_{j-1})$, $deg(g_j) \leq d$, sample $r_j \in \mathbb{F}$, send r_j,

P: compute $g_n(x) = g(r_1 \dots r_{n-1}, x)$, send $g_n(x)$,
V: checks $g_n(0) + g_n(1) = g_{n-1}(r_{n-1})$, $deg(g_n) \leq d$, for random $r_n \in \mathbb{F}$ check $g_n(r_n) = g(r_1 \dots r_n)$.
Completeness:

✓(sum, all possibilities).

Cost:

Prover: $O(2^n)$,

verifier: evaluate g_i $\forall i$, g at one point, $<< O(2^n)$.

Communication:

$deg(g_1) + \dots + deg(g_{n-1}) + O(n)$ elements of $\mathbb{F}$.

Prove that $H_g = \sum_{b_1 \dots b_n \in \{0,1\}} g(b_1 \dots b_n)$.
Soundness:

P: sends $g_i(x)$.

If P cheats, at least one of polynomials is not correct.

Sends $g_i^{'}(x) \neq g_i(x)$.

Verifier checks $g_i^{'}(r_i) = g_{i+1}(0) + g_{i+1}(1) = g_i(r_i)$

$\rightarrow$ probability of this: $\leq \frac{d}{|\mathbb{F}|}$.

Soundness error: $\leq n \cdot \frac{d}{|\mathbb{F}|}$; union bound of rounds.

Application 1:

Counting solutions of SAT.

F SAT formula with s operations and n variables.

Replace with polynomial $g(x_1 \dots x_n)$ such that $F(b_1 \dots b_n) = g(b_1 \dots b_n)$.

For every $b_1 \dots b_n$:

replace $AND(x,y)$ with $x \cdot y$, $OR(x,y)$ with $x + y - x \cdot y$, $NOT(x)$ with $1 - x$.

Number of SAT solutions $= \sum_{b_1 \dots b_n \in \{0,1\}} F(b_1 \dots b_n)$
$= \sum_{b_1 \dots b_n \in \{0,1\}} g(b_1 \dots b_n)$.

Prover can prove that $H_g =$ number of solutions by using sum-check.

Complexity:

prover: $O\left(2^n\right)$,

proof size (communication complexity): $O(n)$ - n polynomials,

verifier: $O(n + s)$.

Error: $\leq \frac{n \cdot s}{|\mathbb{F}|}$; s: number of operations.

Application 2:

Counting triangles in G.

A: adjacency matrix.

Number of triangles in $G = \frac{tr\left(A^3\right)}{6}$.

We think of A as a mapping.

$[n] \times [n] \to \{0, 1\}$.

Define $A' : \{0,1\}^{\log_2 n} \times \{0,1\}^{\log_2 n} \to \{0,1\}$,

such that $A(i, j) = A'(binary(i), binary(j))$.

For example: $n = 16$, $A(0,3) = A^{'}(0000, 0011)$.

Now we define polynomial $f_A : \mathbb{F}^{\log_2 n} \times \mathbb{F}^{\log_2 n} \to \mathbb{F}$

$f_A(x_1 \ldots x_{\log_2 n}, y_1 \ldots y_{\log_2 n})$ such that

$f_A(b_1 \ldots c_{\log_2 n}) = A^{'}(b_1 \ldots c_{\log_2 n})$ for every $b_1 \ldots c_{\log_2 n} \in \{0,1\}$.

Example: $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $f_A(x,y) = x(1-y) + y(1-x)$.

In general:

$$\begin{aligned} & f_A(x_1 \ldots x_{\log_2 n}, y_1 \ldots y_{\log_2 n}) \\ = & \sum_{a,b \in \{0,1\}^{\log_2 n}, A^{'}(a,b)=1} (-1)^{num\_zeros(a,b)} (x_1 - (1-a_1)) \ldots (y_{\log_2 n} - (1 - b_{\log_2 n})). \end{aligned}$$

Now we define $g_A(x,y,z) = f_A(x,y) \cdot f_A(y,z) \cdot f_A(z,x)$, $x, y, z \in \mathbb{F}^{\log_2 n}$.

Number of triangles $= \frac{\sum_{a,b,c \in \{0,1\}^{\log_2 n}} g_A(a,b,c)}{6}$

$\implies$ we can use sum-check.

Proof size: $O(3 \cdot \log_2 n)$ (number of rounds $\to$ poly),

verifier: $O(\log_2 n) + O(n^2)$.

12.2 SNARK

Succint Non-interactive ARgument of Knowledge.
Succint: proof short and verification fast.
Non-interactive: just sending a proof.
$x \to P \to V \leftarrow x$.
P: convince V that $f(x) = y$.

- $f(x) = x^3 + x + 5$ as a algebraic circuit.
  Break down into $+, -, *, /$ in some field $\mathbb{Z}_p$
  Proof with states $\overrightarrow{s} = (five, x, out, s_1, s_2, s_3)$.

Example: proof that $f(3) = 35$.
$\overrightarrow{s} = (1, 3, 35, 9, 27, 30)$.

- To R1CS.
  Give vectors $\overrightarrow{a_i}, \overrightarrow{b_i}, \overrightarrow{c_i}$ for each state such that

  $$(\overrightarrow{a_i} \cdot \overrightarrow{s}) \cdot (\overrightarrow{b_i} \cdot \overrightarrow{s}) = (\overrightarrow{c_i} \cdot \overrightarrow{s}) \iff \text{gate } i \text{ was correctly calculated.}$$

  Example.

  For gate 1 $(\cdot)$:

  $$\begin{aligned} \overrightarrow{a_1} &= [0, 1, 0, 0, 0, 0] \\ \overrightarrow{b_1} &= [0, 1, 0, 0, 0, 0] \\ \overrightarrow{c_1} &= [0, 0, 0, 1, 0, 0] \end{aligned}$$

  $\overrightarrow{a_1} \cdot \overrightarrow{s} = x, \overrightarrow{c_1} \cdot \overrightarrow{s} = s_1$.

  For gate 3 $(+)$:

  $$\begin{aligned} \overrightarrow{a_3} &= [0, 1, 0, 0, 1, 0] \\ \overrightarrow{b_3} &= [1, 0, 0, 0, 1, 0] \\ \overrightarrow{c_3} &= [0, 0, 0, 0, 0, 1] \end{aligned}$$

  $(x + s_2) \cdot 1 = s_3$.

  We have instead of circuit

  $$\begin{bmatrix} \overrightarrow{a_1} \\ \overrightarrow{a_2} \\ \vdots \\ \overrightarrow{a_n} \end{bmatrix} \cdot \overrightarrow{s} \odot \begin{bmatrix} \overrightarrow{b_1} \\ \overrightarrow{b_2} \\ \vdots \\ \overrightarrow{b_n} \end{bmatrix} \cdot \overrightarrow{s} - \begin{bmatrix} \overrightarrow{c_1} \\ \overrightarrow{c_2} \\ \vdots \\ \overrightarrow{c_n} \end{bmatrix} \cdot \overrightarrow{s} = \overrightarrow{0}.$$

  $\odot$: coordinate-wise multiplication.
  $\overrightarrow{s}$ needs to be solution for

  $$A \cdot \overrightarrow{s} \odot B \cdot \overrightarrow{s} = C \cdot \overrightarrow{s};$$

  A, B, C $m \times n$ matrices.

- To Quadratic Arithmetic Programs (QAP).

  Let $a_i(x)$ be a polynomial such that

  $a_i(j) = \overrightarrow{a}_j[i]$ for $i \in [n], j \in [m]$.

  $A = \begin{bmatrix} a_1(1) & a_2(1) & \dots & a_n(1) \\ a_1(2) & \dots & & \\ \vdots & & & \\ a_1(m) & \dots & & a_n(m) \end{bmatrix}.$

  Example:

  $$a_1(x) = -5 + 9.16x + 5x^2 + 0 - 833x^3$$

  $$a_2(x) = 8 - 11.33x + 5x^2 - 0.666x^3$$

  $$a_3(x) = 0$$

  $$\vdots$$

  $$a_6(x) = \dots$$

  $$[a_1(1) \dots a_6(1)] = [0, 1, 0, 0, 0, 0] = \overrightarrow{a_1}.$$

  We get a_i with interpolation $deg\ a_i \leq n - 1$.

  $$([a_1(x), a_2(x) \dots a_n(x)] \cdot \overrightarrow{s}) \odot ([b_1(x), b_2(x) \dots b_n(x)] \cdot \overrightarrow{s}) - ([c_1(x), c_2(x) \dots c_n(x)] \cdot \overrightarrow{s})$$

  should have zeros in $1, 2 \dots m$

  $\iff \quad A(x) \cdot B(x) \cdot C(x) = (x-1)(x-2) \dots (x-m) \cdot h(x)$.

  Summary up to now:

  – instead of states we have polynomials,

  – instead of states, we have coefficients $\cdot \overrightarrow{s}$.

  $a_i(x), b_i(x), c_i(x) \to P \to V \leftarrow a_i(x), b_i(x), c_i(x)$.

  $P \to V$: $A(x), B(x), C(x), h(x)$: too much.

  $P \to V$: $A(r), B(r), C(r), h(r)$, r random.

  V: checks $A(r) \cdot B(r) = C(r) + h(r) \cdot t(r)$;

  works if V doesn't cheat.

  Cryptographic background:

- Lets have pairs
  $(g_1, h_1), (g_2, h_2) \dots (g_n, h_n)$, where $g_i^k = h_i$, you don't know k.
  Cryptographic assumption: if we provide (g', h') such that
  $(g')^k = h'$, then $g' = g_1^{k_1} \cdot g_n^{k_n}, h' = h_1^{k_1} \cdot h_n^{k_n}$.
- Pairing groups.
  In some group one can define a pairing
  $e : G \times G \to G_r$ such that
  $e(g_1 \cdot g_2, h) = e(g_1, h) \cdot e(g_2, h)$,
  $e(g, h_1 \cdot h_2) = e(g, h_1) \cdot e(g, h_2)$,
  $e(g^x, g^y) = e(g, g)^{xy}$.

Assume P, V have

- $g^{a_1(r)}, g^{a_2(r)} \dots,$
- $g^{b_1(r)}, g^{b_2(r)} \dots,$
- $g^{c_1(r)}, g^{c_2(r)} \dots,$
- $g^{t(r)},$
- $g, g^r, g^{r^2} \dots g^{r^{n-1}}$

without knowing r.
Improved protocol:
P sends to V

- $g^{A(r)} = \left(g^{a_1(r)}\right)^{k_1} \dots \left(g^{a_n(r)}\right)^{k_n}$
- $g^{B(r)} \dots$
- $g^{C(r)} \dots$
- $g^{h(r)} = g^{h_0} \cdot (g^r)^{k_1} \dots \left(g^{r^{n-1}}\right)^{k_{n-1}}.$

V: checks $e\left(g^{A(r)}, g^{B(r)}\right) = e\left(g^{C(r)}, g\right) \cdot e\left(g^{h(r)}, g^{t(r)}\right)$.
Problem: $g^{A(r)}$ needs to be linear combination of $g^{a_1(r)} \dots g^{a_n(r)}$, also $g^{B(r)}, g^{C(r)}$.

We need additional values:

- $g^{a_1(r)\cdot k_1} \dots g^{a_n(r)\cdot k_1}$, k_1 unknown,
- $g^{b_i(r)\cdot k_2} \dots$,
- $g^{c_i(r)\cdot k_3} \dots$,
- $g^{k_1}, g^{k_2}, g^{k_3}$.

Prover also submits:

- $g^{k_1 A(r)} = \left(g^{a_1(r)k_1}\right)^{s_1} \dots \left(g^{a_n(r)k_1}\right)^{s_1}$
- $g^{k_2 B(r)} = \dots$
- $g^{k_3 C(r)} = \dots$

Verifier calculates

- $e\left(g^{A(r)}, g^{k_1}\right) = e\left(g^{k_1\cdot A(r)}, g\right)$,
- $e\left(g^{B(r)}, g^{k_2}\right) = \dots$

$\implies$ by crypto assumption $A(r)$ is linear combination of $a_1(r) \dots a_n(r)$.
Downside: we need $g^{a_1(r)} \dots$